НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено
Завідувач кафедри
_____ О.В. Коваль
(підпис) (ініціали, прізвище)
“___”_____2019р.

ДИПЛОМНА РОБОТА

на здобуття ступеня бакалавра

з напряму підготовки
6.050103 “Програмна інженерія”

на тему___Система автоматизованого бронювання квитків у кінотеатрі з___
_____використанням ОС Android_____
_____
_____

Виконав (-ла): студент (-ка) ___4__курсу, групи__ТВз-31__

_____Осередчук Михайло Михайлович_____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник _____к.т.н., доцент Кублій Л. І._____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент _____
_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший, бакалаврський

Напрям підготовки 6.050103 “Програмна інженерія”

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.В. Коваль
(підпис)

”___”_____2019 р.

ЗАВДАННЯ

на дипломну роботу студенту

Осередчуку Михайлу Михайловичу
(прізвище, ім’я, по батькові)

1. Тема роботи Система автоматизованого бронювання квитків у кінотеатрі з використанням ОС Android

керівник роботи к.т.н., доцент Кублій Лариса Іванівна
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від 22.05.2019 р. № 1324-С

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи мова програмування Java, середовище розробки Android Studio, система керування базами даних FireBase

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз задачі розробки програмного забезпечення для бронювання квитків у кінотеатрі зі смартфона. Аналіз та оцінка існуючих рішень. Вибір програмних засобів реалізації програмного рішення. Розробка програмного забезпечення для перегляду деталей та бронювання квитків з використанням засобів мобільного зв’язку.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)___ схема потоків інформації в програмному комплексі; схема процесу аналізу знімка і прийняття рішення; графічний інтерфейс комплексу_____
_____

6. Дата видачі завдання " 11 " жовтня 2016 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Вивчення та аналіз задачі	01.02.2017 — 31.03.2019	
2.	Розробка архітектури та загальної структури системи	01 — 21.04.2019	
3.	Розробка структур окремих підсистем	22 — 28.04.2019	
4.	Програмна реалізація системи	29.04 — 13.05.2019	
5.	Захист програмного продукту	14.05.2019	
6.	Оформлення пояснювальної записки	15.05 — 01.06.2019	
7.	Передзахист	28.05.2019	
8.	Захист	18.06.2019	

Студент _____ ____Осередчук М.М._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ _____Кублій Л. І._____
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

Метою роботи було створення програми для перегляду деталей фільмів, що транслюються в кінотеатрі, вільних місць в залі та бронювання вибраних місць. Проект створено в сучасному середовищі розробки Android Studio, мова програмування — Java.

Записка містить 75 сторінок, 31 рисунок, 20 посилань і 3 додатки.

Ключові слова: СИСТЕМА БРОНЮВАННЯ, МОБІЛЬНІ ЗАСОБИ, ОС ANDROID, ANDROID STUDIO, JAVA.

ANNOTATION

The purpose of the work was to create a program to view the details of films broadcasting in the cinema, free seats in the hall and reservation of selected places. The project created in a modern development environment Android Studio, programming language — Java.

The note contains 75 pages, 31 images, 20 references and 3 applications.

Keywords: RESERVATION SYSTEM, MOBILE MEANS, OS ANDROID, ANDROID STUDIO, JAVA.

ЗМІСТ

ВСТУП

У наш час спостерігається тенденція щодо розширення ринку надання різноманітних розважальних послуг. До цих послуг, звичайно ж, потрібно віднести й перегляд фільмів у кінотеатрах. Можна помітити, що кількість кінотеатрів невпинно зростає як у великих містах з населенням більше мільйона, так і в маленьких містах. Не дивлячись на це, є певні й незмінні лідери у наданні цієї послуги.

Для того, щоб виграти такі “перегони” у конкурентів, компанії потрібно реалізовувати основні гілки стратегічного зростання й покращення своєї компанії чи навіть мережі.

Без сумніву, по-перше, — це збільшення своєї частини на ринку: вихід у міста-мільйонники, обласні центри, де помічається нестача сучасних кінотеатрів, і збільшення свого знаходження в усіх регіонах.

По-друге, — це розробка й реалізація найпотрібнішої на ринку структури багатозального кіноцентру для зручності відвідувачів цього закладу, щоб надати широкий репертуар і можливість протягом короткого часу потрапити на вподобаний фільм.

По-третє, оптимізація й покращення пристроїв і функцій мережі, що передбачає оцінку економічних показників підприємства, змінену його пристроїв і діяльності.

Процес автоматизації роботи кінотеатру полягає в розробці та впровадженні програмного забезпечення для продажу й автоматичного аудиту квитків, враховуючи різні типи місць у кінотеатрі, цінову політику, програму підтримки постійних клієнтів, систему знижок та інші акції. Процес автоматизації неодмінно зв’язаний з оновленням не тільки програмного продукту, а й з оновленням і покупкою нового обладнання і витратою грошей на його обслуговування. Сюди потрібно включити комп’ютер кожному продавцеві чи касирові, сервер, принтер, касові апарати, а також інше устаткування таке, як комутатори.

Використання системи має спростити управління всіма процесами, пов’язаними з прийманням та обробкою замовлень, надати можливість менеджеру вчасно отримати потрібну інформацію і після цього будувати вірну економічну політику мережі.

Позитивний ефект від продуманого застосування автоматизованих систем управління в кіноцентрах безумовний. В умовах фінансової кризи інформаційні технології повинні стати серйозним інструментом для оптимізації системи управління, зменшення витрат і надати безперечні переваги над конкурентами. Автоматизація процесів — запорука хорошого управління. Автоматизація призводить до зменшення довгих операцій, дає можливість швидше обслуговувати глядачів, надає більше можливостей для контролю, при цьому значно покращується робота щодо закупівель і постачань тощо. Все це в свою чергу значно збільшує прибуток, товарообіг, зменшує витрати. Зменшення рутинних операцій значно сприяє скороченню витрат на техніку й персонал.

Автоматизація кінотеатру чи мережі — сьогодні така ж потреба, як і встановлення екрану. Якісний звук і зображення, зручні місця повинні поєднуватися зі злагодженою роботою автоматизованої системи з продажу та резервування квитків. Сучасний кінопоціновувач вже відвик від таких ситуацій, як продаж квитків різним людям на одне і те ж місце. Йому теж важко уявити, що квитки на фільм не можна забронювати за телефоном або “онлайн”.

Розроблена в дипломній роботі програма Kino Booker призначена для користувачів смартфонів і забезпечує швидке й просте бронювання чи покупку квитків.

Завдяки використанню сучасних технологій розроблене програмне забезпечення має не складний і зрозумілий інтерфейс, швидко працює й надає актуальну інформацію.

Середовищем розробки програмного забезпечення вибрано Android Studio 3.4, мовою програмування — Java. Для створення бази даних і роботи з нею — Firebase (від Google). Програма працює на смартфонах з операційною системою Android версій не нижче Android 5.0 Lollipop. Тестування програми виконано на емуляторі

Android Virtual Device (AVD) з версією Android 7.0, а також на смартфоні з версією Android 9.0.

Розроблене програмне забезпечення призначене для звичайних користувачів.

У першому розділі пояснювальної записки сформульовано постановку задачі. У другому — проведено аналіз існуючих програм для бронювання квитків. У третьому розділі обґрунтовано вибір програмних засобів розробки системи. Програмну реалізацію описано в четвертому розділі. У п’ятому розділі продемонстровано методику роботи користувача з мобільним додатком.

1. ЗАДАЧА РОЗРОБКИ СИСТЕМИ АВТОМАТИЗОВАНОГО БРОНЮВАННЯ КВИТКІВ У КІНОТЕАТРІ

Сучасний кіноцентр може запропонувати глядачеві цікаві кіно чи мультфільми, хороше зображення і звук, комфортні сидіння. Але першим місцем у кінотеатрі є звичайна квиткова каса, в якій можна купити квитки.

Для того, щоб не стояти у чергах чи купити найкраще місце, людині зручніше просто скористатися мобільним додатком. Там є вся інформація про фільм і відображаються вільні місця в залі з наступною можливістю забронювати місце чи купити квиток.

Метою дипломної роботи є розробка й створення на основі сучасних комп'ютерних технологій програмного продукту для перегляду інформації про фільми та вільні місця на них з можливістю забронювати місця у кінотеатрі і придбати квиток з використанням мобільних засобів зв'язку.

Розроблюваний мобільний програмний додаток повинен забезпечувати виконання таких функцій:

— надати користувачеві інформацію про доступні фільми;

— показати опис фільму, постер, трейлер, рейтинг;

— показати доступні місця з можливістю вибору одного чи кількох місць;

— швидко обмінюватися даними з базою даних;

— надати можливість забронювати квиток та обновити інформацію;

— надати можливість купити квиток;

— прийняти інформацію від користувача (контактні дані, дані карти);

— надіслати інформацію про успішне бронювання чи покупку квитка.

Для роботи програми потрібен смартфон з операційною системою Android і база даних, яка повинна бути постійно підключена до мережі Інтернет.

Потенційними користувачами створюваного продукту є звичайні користувачі, які мають смартфон з операційною системою Android.

2. АНАЛІЗ ІСНУЮЧОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

На сьогодні, на жаль, представлена мала кількість мобільних застосунків для онлайн-бронювання та купівлі квитків у кінотеатрах. Більшість сервісів представлені як веб-сайти, а Google Play Market є незначна кількість. Кожен з них має переваги й недоліки.

Нижче проаналізовано сервіс пошуку й бронювання квитків vkino [1, 2], Android-додатки “Планета Кіно” [3] і “SmartCinema” [4].

2.1 Сервіс онлайн-продажу квитків у кіно “vkino”

Сервіс представлений веб-сайтом [1], на якому потрібно перш за все обрати своє місто. Вигляд ресурсу подано на рисунку 2.1.

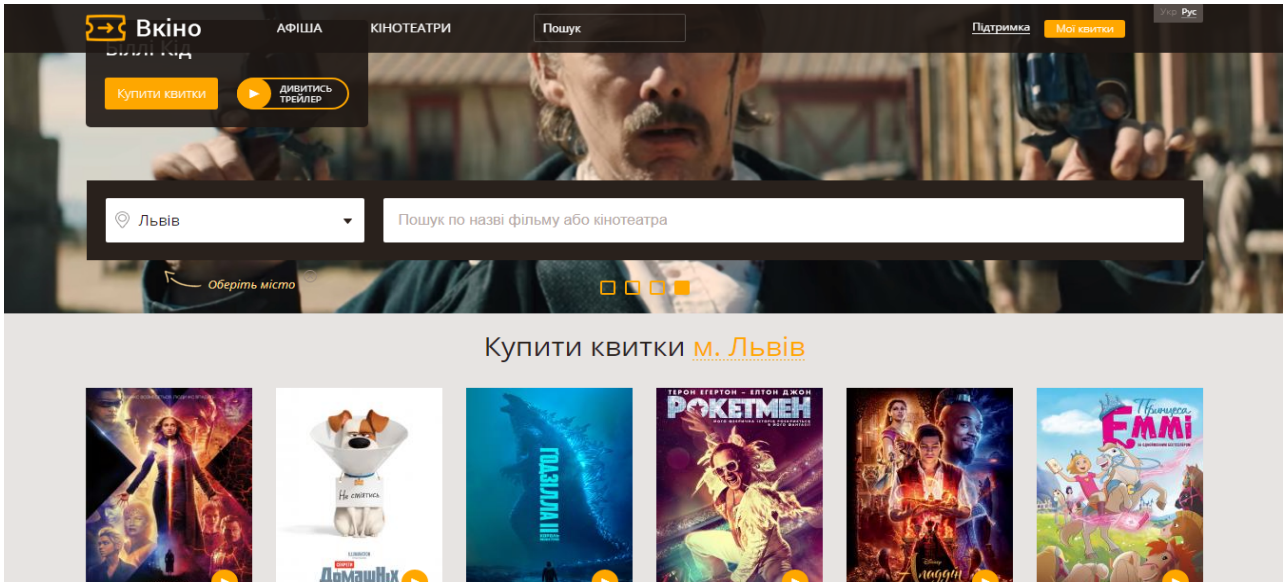

Рисунок 2.1 — Вигляд веб-сайту vkino, головна сторінка

Сервіс має такий функціонал:

— пошук міста;

— пошук фільму;

— пошук кінотеатру;

— бронювання квитка;

— купівля квитка;

— повернення квитка;

— перегляд всієї інформації про фільм.

Цей сервіс також представлений додатками для мобільних операційних систем IOS та Android. Розглянемо останній. Тут збереглися всі функції основного сайту. Завдяки додатку можна:

— знайти фільм;

— знайти кінотеатр;

— купити чи забронювати квиток.

Недоліками сервісу є:

— обов’язкова реєстрація (рисунок 2.2);

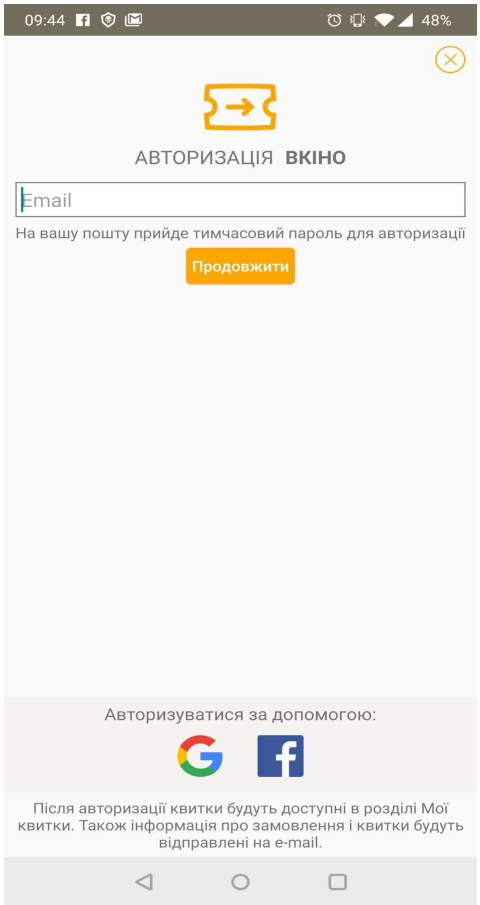

Рисунок 2.2 — Обов’язкова реєстрація для бронювання квитків

— комісія за бронювання чи купівлю квитка.

Перевагами сервісу в цілому є:

— найбільший вибір міст та кінотеатрів;

— можливість повернути квитки.

2.2. Мобільний додаток “Планета Кіно”

Ще одним великим сервісом, представленим на Android, є “Планета кіно” [3]. Вигляд додатку подано на рисунку 2.3.

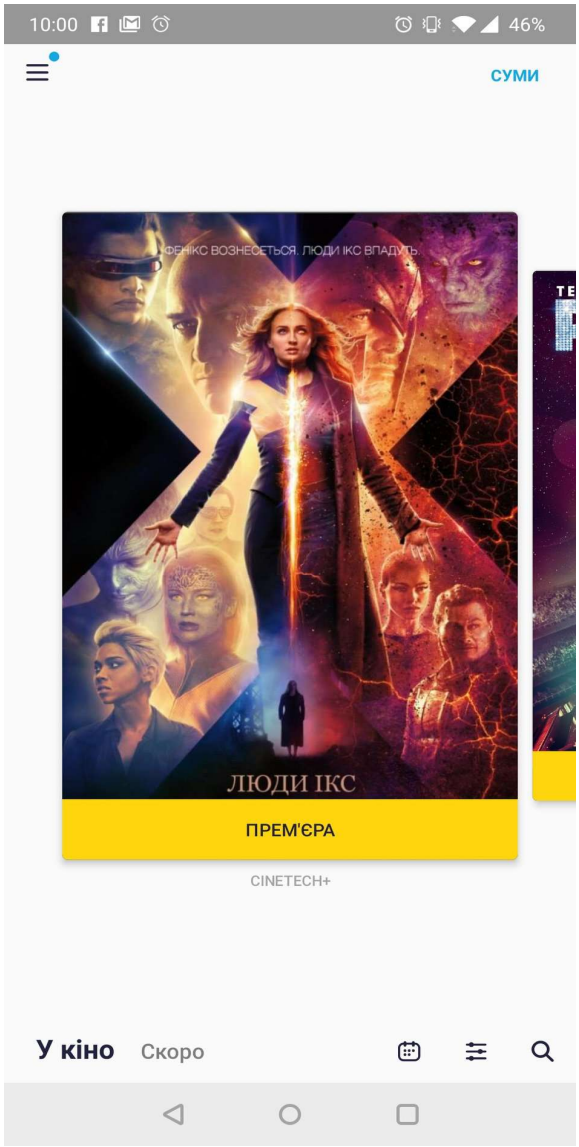

Рисунок 2.3 — Вигляд головного меню додатку “Планета кіно”

У сервісі “Планета кіно” доступні такі функції:

— вибір міста, фільму й сеансу;

— перегляд повної інформації про фільм;

— купівля квитка.

Перевагами додатку є можливість пошуку за фільмом.

Недоліками є:

— повільна робота додатку;

— обов’язкова реєстрація.

2.3. Мобільний додаток “SmartCinema”

Ще одним з небагатьох додатків, які можна знайти в Play Market, є “SmartCinema” [4]. Загальний вигляд додатку подано на рисунку 2.4.

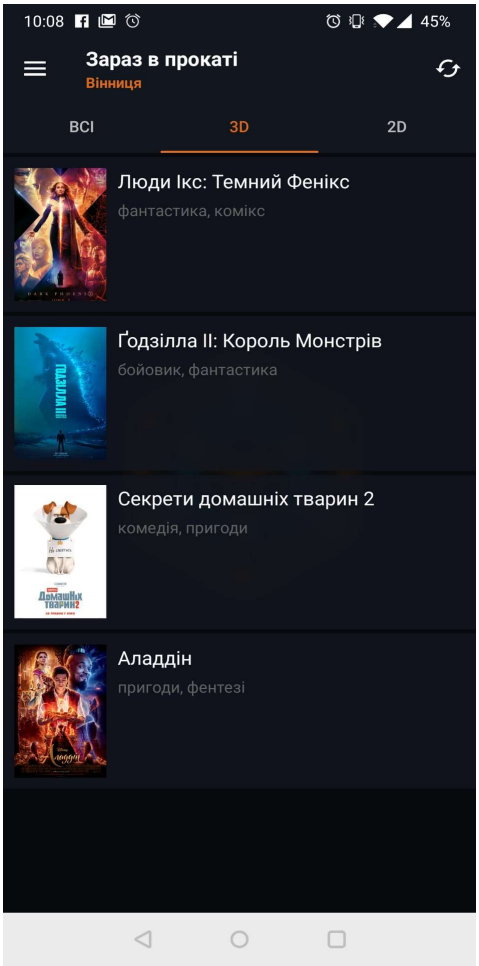

Рисунок 2.1.4. Вигляд додатку “SmartCinema”

Додаток “SmartCinema” дає можливість обрати кінотеатр, режим перегляду (2D, 3D), фільм і сеанс.

Недоліками цього додатку є:

— обов’язкова реєстрація навіть для того, щоб переглянути вільні місця;

— комісія за покупку квитка.

2.4. Необхідність розробки альтернативи

Було розглянуто кілька сервісів бронювання та купівля квитків у кінотеатрі. Брався до уваги функціональний набір додатків, було проаналізовано переваги й недоліки кожного з них.

Головною необхідністю розробки системи бронювання квитків Kino Booker є те, що подібних рішень для ОС Android не так вже й багато, тому є затребуваним створення альтернативи.

Основними недоліками розглянутих програмних засобів є:

— комісія за бронювання чи покупку квитка (платно);

— відсутність звичайної броні, тільки покупка;

— обов’язкова реєстрація.

Основними перевагами розробленого продукту є:

— швидкість роботи;

— відсутність обов’язкової реєстрації;

— безкоштовність.

Таким чином, проблема розробки нових програм для бронювання квитків у кінотеатрі на ОС Android залишається актуальною.

3. ОБҐРУНТУВАННЯ ВИБОРУ ЗАСОБІВ РЕАЛІЗАЦІЇ ПРОГРАМНОГО КОМПЛЕКСУ

Для розробки програмного продукту програміст має обрати такі технології й засоби, які б спростили йому його роботу, надаючи необхідні інструменти для виконання завдання. Правильний підхід до вибору технологій і засобів програмної реалізації безпосередньо впливає на час розроблення, якість, швидкість роботи продукту та його якості.

Для розробки системи бронювання Kino Booker було вибрано такі засоби:

— операційна система Android

— середовище розробки програми — Android Studio 3.4 (від Google)

— мова програмування — Java;

— для роботи з базами даних — сервіс Firebase (від Google).

Система бронювання працює на смартфонах з операційною системою Android версій не нижче Android 5.0 Lollipop.

Тестування системи бронювання виконано на емуляторі Android-смартфонів AVD, а також на смартфоні з версією Android 9.0.

3.1. Середовище розробки Android Studio

Середовище розробки Android Studio [5] — інтегроване середовище розробки (IDE) для платформи Android.

Це офіційне середовище розробки Android-програм від компанії Google для відповідної операційної системи, яке замінило плагін ADT для середовища Eclipse і є зараз одним із найпопулярніших інструментів серед розробників. Версія Android Studio v.3.4 була випущена 17 квітня 2019 року.

Дане середовище є абсолютно безкоштовним для завантаження й користування. Тут наявні макети для створення користувацького інтерфейсу,

завдяки чому воно переважно і використовується при розробці застосунків для Android. У цьому середовищі є всі інструменти не тільки для розробки програм для планшетів, смартфонів, персональних комп'ютерів, а й для новіших рішень таких, як розумні телевізори (Android TV), годинники (Android Wear), автомобілі (Android Auto) та інші додаткові модулі.

Розробки у середовищі Android Studio є дуже гнучкими. Це реалізовано за допомогою відображення всіх робочих файлів в одній програмі в одному вікні. Корисною функцією є можливість побачити всі візуальні зміни розроблюваної програми в режимі реального часу.

Середовище Android Studio надає можливість протестувати свій проект на різних пристроях як підключаючи смартфон, так і запустивши віртуальну машину прямо із середовища розробки.

Вигляд середовища Android Studio подано на рисунку 3.1.

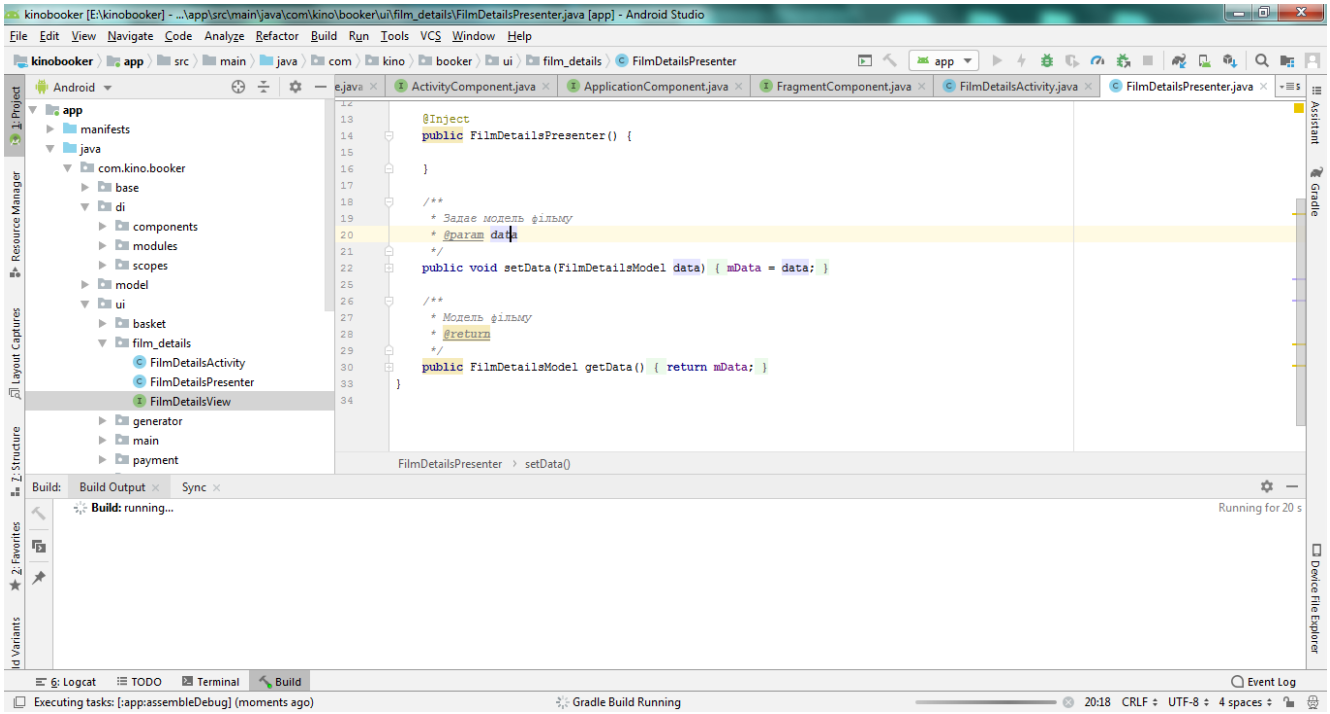

Рисунок 3.1 — Вигляд середовища Android Studio версії v3.4

У вбудованому емуляторі Android-пристроїв можна встановити різні технічні конфігурації, версії Android і роздільну здатність екрану. Також можна отримати інформацію про рівень приблизної продуктивності для конкретного пристрою.

Середовище розробки адаптоване для виконання завдань, які вирішуються у процесі розробки Android-додатків. У середовище включені засоби для спрощення розробки й тестування програм. Можна перевірити проект на сумісність з різноманітними пристроями такими, як смартфони, планшети, персональні комп'ютери, розумні годинники та інші.

У середовищі є функції для завантаження прикладів коду з популярного ресурсу GitHub.

Перевагами даного середовища розробки є такі:

— підтримка багатьох мов програмування таких, як Java, C, C++, Kotlin [6];

— доступність всіх функцій Java 7, 8 і багатьох функцій новіших релізів 9-12 [7, 8];

— зручне редагування коду;

— багатофункціональна консоль розробника;

— можливість одразу із середовища запустити емулятор чи підключити пристрій для тестування;

— готові шаблони для створення макетів, дизайнів та компонентів Android;

— інтеграція з із сервісом управління базами даних Firebase;

— інтеграція з GitHub;

— вбудована функція для аналізу вмісту додатку (APK Analyzer);

— інструменти для тестування програм;

— безкоштовність;

— єдиний великий форум розробників;

— постійне оновлення, отже, можливість розробки для найсвіжіших версій OS Android.

Недоліками середовища Android Studio є:

— для роботи потрібен потужний комп'ютер;

— неможливість розробки серверної частини проектів для комп'ютера та смартфонів.

— не надто багатий функціонал персоналізації.

Згідно з даними офіційного сайту [9] системними вимогами є такі:

— версія OS (Android-студія була встановлена на персональних комп'ютерах з OS Windows) — Microsoft Windows 10/8/7/Vista/2003 (32 або 64-bit);

— оперативна пам'ять — 3 ГБ (мінімум), 8 ГБ (рекомендовано); +1 ГБ для Android Emulator;

— вільне місце на диску — 2 ГБ мінімум (500 МБ для IDE + 1.5 ГБ для Android SDK і образу системи емулятора), 4 ГБ SSD рекомендовано;

— версія JDK — Java Development Kit 8;

— роздільна здатність екрану — 1280 x 800 (мінімум).

Переглянувши системні вимоги, можна зробити висновок, що для розробки мобільного додатку необхідно використати потужний персональний комп'ютер.

На сьогодні більшість проектів для Android створюється за допомогою середовища Android Studio. Оскільки середовище надає різноманітний і широкий функціонал, підтримує популярні мови програмування такі, як Java, C/C++ тощо, то це середовище є дуже популярним як серед новачків, так і серед професіоналів.

Середовище Android Studio по праву вважається найкращим безкоштовним інтегрованим середовищем розробки для Android-програмістів.

Також є ще середовище IntelliJ IDEA, але воно в основному використовується для проектів не тільки для Android.

3.2. Мова програмування Java

Мова Java — об'єктно-орієнтована мова програмування [10]. Саме тому для неї притаманні три парадигми об'єктно-орієнтованого програмування:

— наслідування;

— інкапсуляція;

— поліморфізм.

Мова програмування Java була розроблена компанією Sun Microsystems. Програми, написані мовою Java, переважно компілюються в спеціальний байт-код, тому вони можуть працювати на будь-якій віртуальній Java-машині (JVM) незалежно від комп'ютерної архітектури. Дата офіційного релізу — 23 травня 1995

року. Сьогодні технологія Java надає можливості для перетворення сталих Web сторінок в динамічні інтерактивні документи, а також для створення окремих додатків, що не є залежними від платформи.

Мова Java запозичила синтаксис із C і C++ [11]. Не пов'язуючи розробку з конкретною платформою, розробник мови Java Ґослінг почав з розширення компілятора C++. Згодом він зрозумів, що мова C++, як її не розширюй, не зможе задовільнити всі потреби. Тому в середині 1991 року було розроблено мову Oak (згодом при пошуку торгової марки її назва була замінена на Java) [12]. Гослінг вважав браузер важливим компонентом "створення ринку" для додатків, серверів і середовищ розробки. У цих сервісах мова Java відіграє ключову роль. До появи Java веб-сторінка фактично представляла собою листок паперу. З появою Java браузер задає структуру і значно розширює можливості.

Цілі розробки мови Java:

— доведення синтаксису мови до звичного, простого та об'єктно-орієнтованого;

— безпека та безвідмовність реалізації;

— збереження переносимості на незалежності від архітектури;

— якомога вища продуктивність;

— багатопоточність, інтерпретація, динамічність зв'язування модулів.

Маючи такі інструментальні засоби як Java і Web, можна отримати з нуля організовану систему.

Багато компаній організовують бази даних у вигляді Web-сторінок з використанням інтерфейсу Common Gateway Interface (CGI) — специфічного стандарту для роботи зовнішніх програм на сервері HTTP.

Головні можливості Java:

— автоматичне управління пам'яттю;

— додаткові можливості для оброблення окремих ситуацій;

— великий набір методів фільтрування вводу та виводу;

— перелік поширених колекцій, таких як стек, масив, список тощо;

— присутність простих методів та засобів для побудови веб-застосунків;

— присутність класів, що забезпечують http-запити та обробку відповідей на них;

— вбудовані в Java засоби для побудови багатонитевих застосунків;

— об’єднаний доступ до бази даних: на рівні окремих SQL-запитів — на основі JDBC, SQLJ; на рівні концепції об’єктів, що мають здатність до зберігання в базі даних — на основі Java Data Objects і Java Persistence API;

— наявність різноманітних шаблонів;

— паралельна робота багатьох програм.

У набір мови Java входить:

— інтегроване середовище розробки Integrated Development Environment (IDE) — інструменти, які допомагають запускати, редагувати і компілювати код. Найпопулярніші з них — IntelliJ IDEA, Eclipse і NetBeans;

— виконуюча система Java Runtime Environment (JRE). Механізм розповсюдження програмного забезпечення, складається з автономної віртуальної машини Java, стандартної бібліотеки Java (Java Class Library) і інструментів налаштувань;

— комплект розробника Java Development Kit (JDK). За допомогою JDK і стандартного блокнота можна писати і запускати / компілювати код мовою Java.

Перевагами мови Java є:

— найбільша серед усіх мов програмування ступінь переносимості програм;

— потужні стандартні бібліотеки;

— вбудована підтримка роботи в мережах (як локальних, так і веб).

Недоліками мови Java є:

— низька порівняно з іншими мовами швидкодія, підвищені вимоги до обсягу оперативної пам’яті;

— великий обсяг стандартних бібліотек і технологій створює складності при вивченні мови;

— постійний розвиток мови викликає наявність як застарілих, так і нових засобів, які мають одне і те ж функціональне призначення.

3.3. Операційна система Android

Система автоматичного бронювання квитків Kino Booker було розроблено під ОС Android. Вибір зупинився на цій операційній системі, оскільки за останніми даними близько 88% усіх смартфонів у світі працюють під ОС Android [13].

Операційна система Android [14] — це розробка компанії Android inc. Зараз вона є власністю компанії Google і використовується на багатьох сучасних пристроях, таких як смартфони, планшети, персональні комп'ютери, телевізори, годинники, автомобільні бортові комп'ютери тощо. Операційна система Android побудована на зміненому ядрі Linux і власній реалізації віртуальної машини Java [15].

Система Android дає можливість створювати Java-застосунки, які керують пристроєм через розроблені Google-бібліотеки.

Операційна система Android працює на віртуальній машині JVM (Java Virtual Machine), яка здійснює виконання команд байт-кодів.

Байт-код — це проміжне рішення, в яке може бути перекладено код програми. Відносно програмного коду, що є читабельним і зручним для користувача (програміста), байт-код — це компактне подання програми, яка є простішою для комп'ютера. Це подання проходить синтаксичний і семантичний аналізи. У ньому в явному вигляді закодовані типи, області видимості тощо. З технічного боку, байт-код є машинно-незалежним кодом, що генерується транслятором і є кодом низького рівня.

Крім використання мови Java, існує можливість створення додатків з використанням мови C++, проте сама компанія Google не рекомендує використовувати її, а лише в певних випадках, наприклад, якщо потрібне низькорівневе налаштування деяких апаратних складових пристрою.

Характеристики ОС Android такі:

— бази даних — SQLlite, mySQL, Firebase тощо;

— інструментарій — VGA, бібліотеки 2D, 3D, розроблені на основі OpenGL ES;

— технології зв’язку — GSM, Bluetooth, EDGE, 3G, 4G (LTE) 5G [16], WiFi, NFC, IrDA тощо:

— браузери — веб-браузер на основі WebKit application framework.;

— обмін даними — CMC, MMC сервіси, VoLTE тощо;

— медіа — MPEG-4, H.264, MP3, AAC, AMR, JPG, PNG, GIF тощо;

— обладнання — смартфони, планшети, нетбуки, телевізори, годинники, відеокамери, фотоапарати, дотикові екрани, компаси, акселерометри, та прискорювачі 3D-графіки.

Переваги ОС Android:

— система Android зарекомендувала себе краще за своїх конкурентів, таких як IOS, Windows Phone mobile, щодо веб-серфінгу, інтеграції з сервісами Google;

— є відкритою платформою, що дає можливість реалізовувати на ній більше функцій;

— в Android-пристроях, як правило, присутній читач карт пам’яті, що робить можливим швидке перенесення файлів з комп’ютера на телефон і навпаки;

— повноцінна реалізація Bluetooth-стека, що забезпечує в тому числі передачу і прийом файлів;

— можливість встановлення додатків зі сторонніх джерел, що дає можливість не використовувати Інтернет-мережу й тестувати додатки;

— доступність на різних апаратних платформах, таких як ARM, MIPS, x86 тощо;

— багатокористувацький режим.

Недоліки ОС Android:

— проблеми з конфіденційністю (встановлені сервіси Google забезпечують можливість передачі ідентифікаційної інформації на сервери компанії, наприклад, інформацію про місцезнаходження);

— безпека (за даними Lookout Security Mobile, за 2011 рік у користувачів Android-смартфонів було вкрадено близько мільйона доларів США [17]);

— операційна система Android не підтримує атрибут download, який використовується в HTML для завантаження файлів з інтернету.

Отже, для розробки додатку було обрано саме ОС Android, як систему, що має відкритий тип, багатий функціонал і є найбільш поширеною. Все це дає можливість без проблем створити й протестувати розроблену програму.

3.4. Сервіс баз даних Firebase

Сервіс Firebase — це безкоштовна платформа для програмування мобільних і веб-додатків. Його створено в 2011 році компанією Firebase Inc. У 2014 році його викупила компанія Google [18, 19].

Вигляд вікна системи Firebase подано на рисунку 3.4.

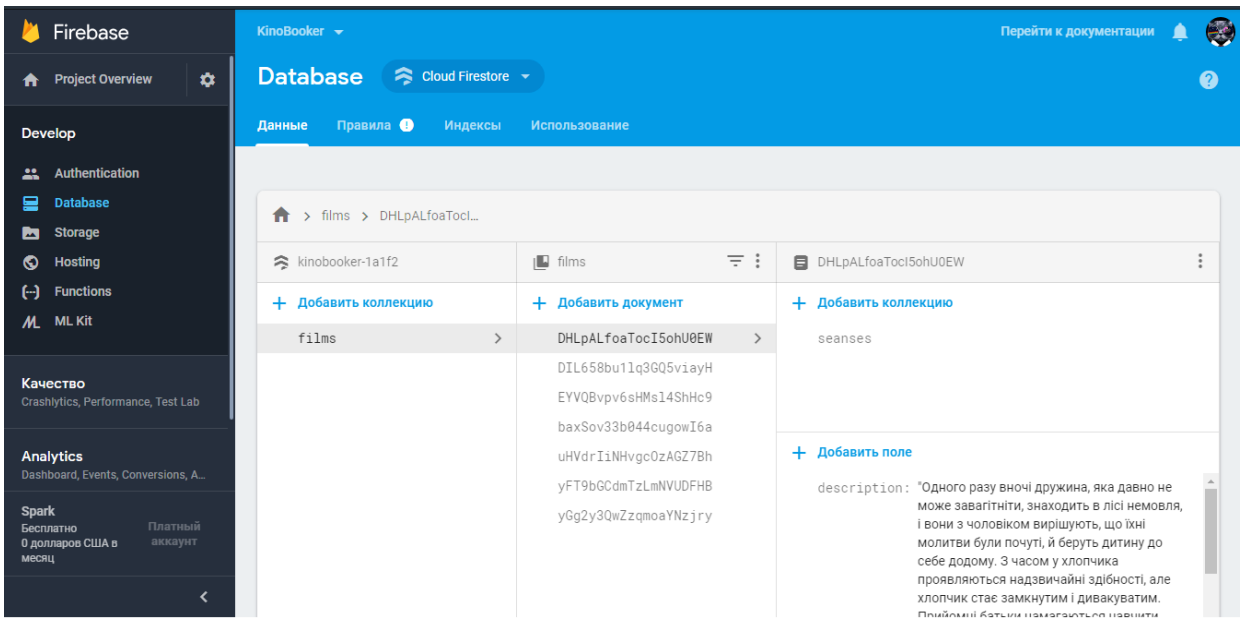

Рисунок 3.4. Вигляд меню з базою даних системи Firebase

Основний сервіс — хмарна СУБД класу NoSQL, що дає можливість розробникам додатків зберігати й синхронізувати дані між кількома клієнтами. Підтримані особливості інтеграції з додатками під операційні системи Android і iOS, реалізовано API для додатків на JavaScript, Java, Objective-C і Node.js, також можливо працювати безпосередньо з базою даних в стилі REST з ряду JavaScript-

фреймворків, включаючи AngularJS, React, Vue.js, Ember.js і Backbone.js. Передбачено API для шифрування даних.

Основні переваги платформи [20]:

— швидкість роботи. У пакеті розробників Firebase були зібрані інтуїтивно зрозумілі API, які спрощують і прискорюють розробку якісних програм;

— готова інфраструктура. Не потрібно створювати складну інфраструктуру або працювати з декількома панелями управління.

— кросплатформеність. Сервіс Firebase працює на будь-яких платформах завдяки пакетам розробки для Android, iOS, JavaScript і C ++. Також можна звернутися до Firebase, використовуючи серверні бібліотеки або REST API;

— можливість масштабування проекту. Якщо додаток стає популярним, то не доведеться змінювати код сервера чи долучати інші ресурси;

— безкоштовість. Більшість функцій Firebase безкоштовні і залишаються такими незалежно від масштабу проектів.

4. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ СИСТЕМИ БРОНЮВАННЯ КВИТКІВ

Систему бронювання квитків у кінотеатрі Kino Booker реалізовано на основі використання і модифікації існуючих і розроблених класів і методів мови програмування Java.

4.1. Архітектура системи

В архітектуру системи Kino Booker бронювання квитків входять такі складові:

— застосунок, встановлений на смартфоні з ОС Android;

— база даних, яка розміщується на хмарному сховищі.

Доступ до застосунку мають усі користувачі, в тому чисті адміністратор.

Доступ до бази даних має тільки адміністратор.

Додаток, встановлений на смартфоні, може передати користувачеві такі дані:

— інформацію про фільм;

— інформацію про сеанси;

— інформацію про вільні й заброньовані місця;

— повідомлення про те, що квиток заброньовано чи куплено.

У базі даних зберігається всі інформація, яка надається користувачеві додатка. Також через базу даних вносяться зміни про деталі фільму, сеансів тощо. Адміністратор може переглядати, які саме місця були заброньовані та зняти броню з них.

Загальну схему потоків інформації системи бронювання Kinoo Booker подано на рисунку 4.1.

Програма, встановлена на телефоні, весь час підключена до бази даних та обмінюється з нею інформацією. Вона передає користувачеві актуальні дані й повертає базі даних зміни, які вніс користувач (броня квитка).

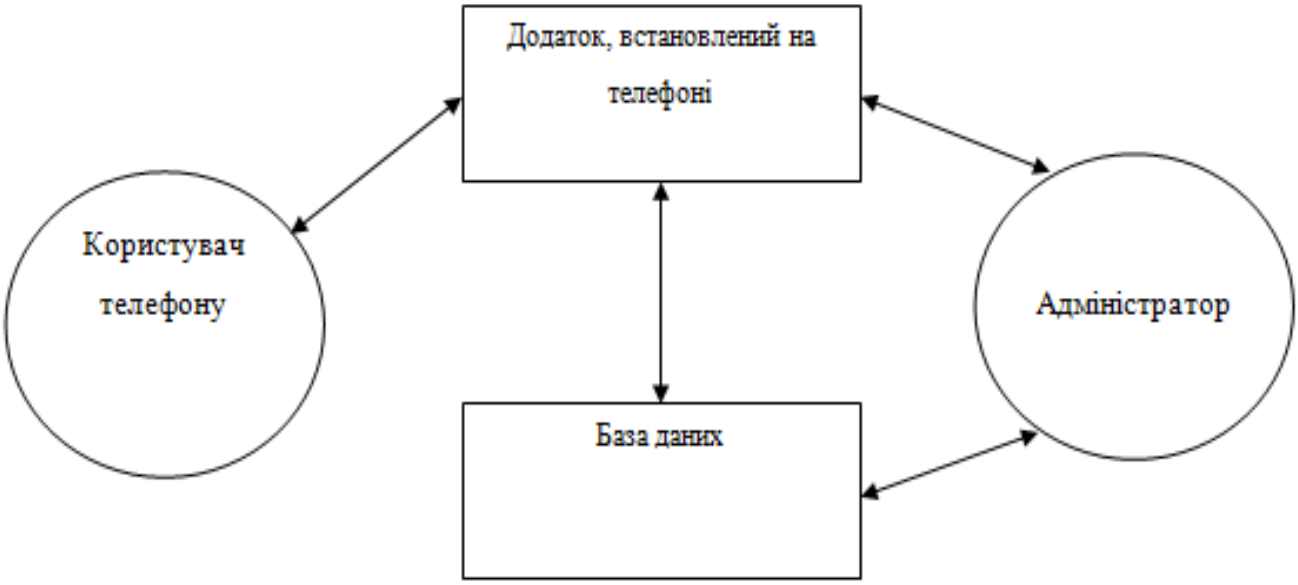

Рисунок 4.1 — Схема потоків інформації у системі

Для адміністратора існує окрема дія — довге натискання, після якого відкривається меню адміністратора, де він може внести зміни в будь-яку частину (фільм, сеанс тощо). Програма передасть ці дані базі даних.

4.2. Класи і методи програми

У розробленій програмній системі бронювання квитків Kino Booker є ряд класів.

Головний клас MainActivity є звичайним класом мови програмування Java, на початку якого йдуть визначення пакета й імпорту зовнішніх пакетів. За замовчуванням він містить тільки один метод onCreate, в якому фактично і створюється весь інтерфейс програми.

У програмі клас MainActivity, крім методу onCreate, містить ще методи onStart, createRecycler, onFilmPressed, onLongPressed.

Методи класу MainActivity подано на рисунку 4.1.

Клас MainPresenter містить метод LoadData, який завантажує з бази даних всю інформацію про фільми.

Методи класу MainPresenter показано на рисунку 4.2.

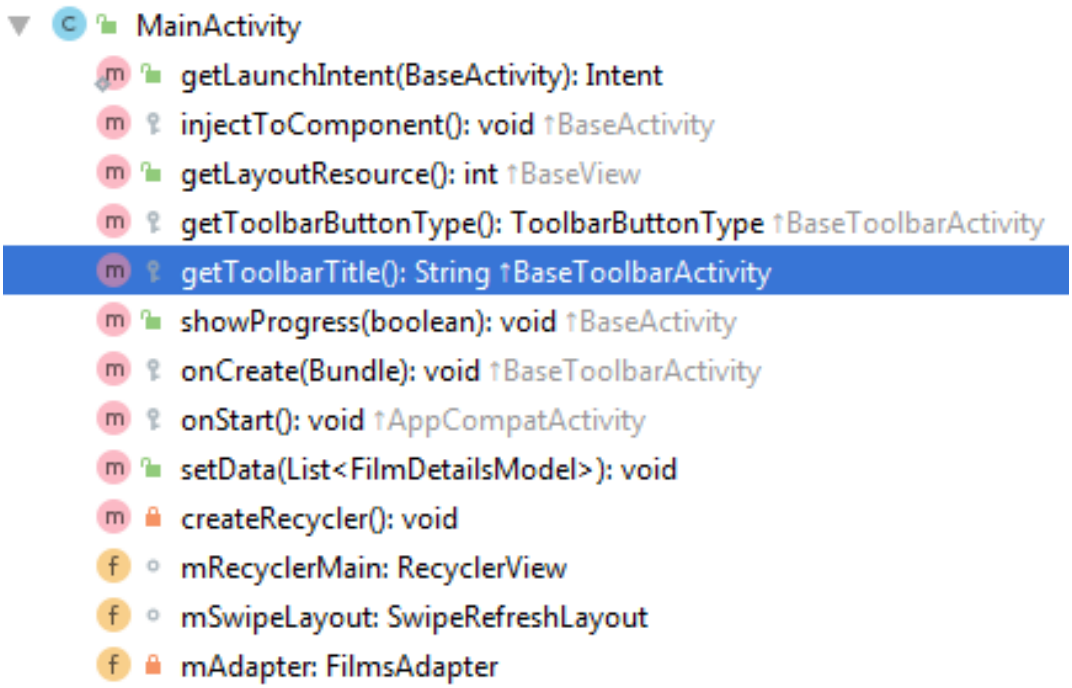

Рисунок 4.1 — Методи класу MainActivity

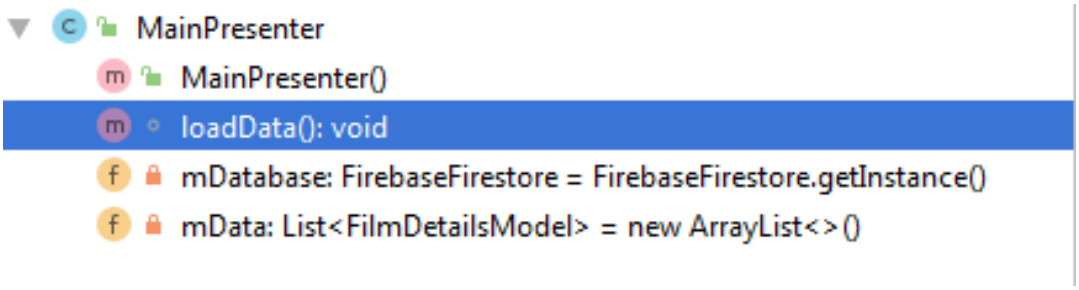

Рисунок 4.2 — Методи класу MainPresenter

Створений додаток для бронювання квитків Kino Booker побудовано за архітектурою MVP (Model, View, Presenter). Тому в програмі є класи моделей, виглядів та презенетрів.

Розглянемо класи моделей.

Клас FilmDetailModel — це клас, що створює модель фільму, тут є такі методи, як FilmDetailsModel, getId, getName, setName, getDescription, setDescription та інші. Ці методи відповідають за подачу інформації про фільм користувачеві створеного мобільного додатка Kino Booker і передачу інформації про фільм від адміністратора до бази даних.

Методи класу FilmDetailModel подано на рисунку 4.3.

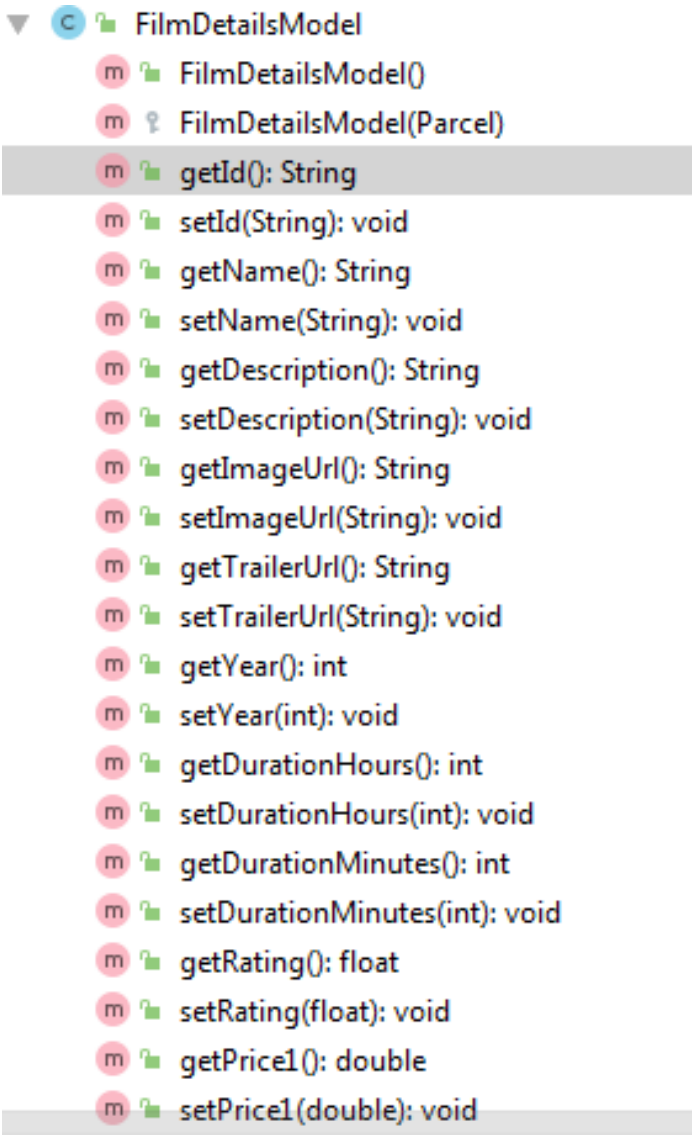

Рисунок 4.3 — Методи класу FilmDetailModel

Клас SeanseModel — тут є всі методи для передачі інформації глядачеві та зчитування даних від адміністратора про сеанси фільму.

Вигляд класу SeanseModel подано на рисунку 4.4.

Клас SeansesListModel — клас, в якому є методи для виводу списку сеансів. Список методів класу SeansesListModel зображено на рисунку 4.5.

Клас SeatModel — це клас, в якому є методи для відображення кінозалу, тобто вигляду місць, перевірка, які є забронювані, а які вільні, розподілу місць за ціновими категоріями.

Методи класу SeatModel зображено на рисунку 4.6.

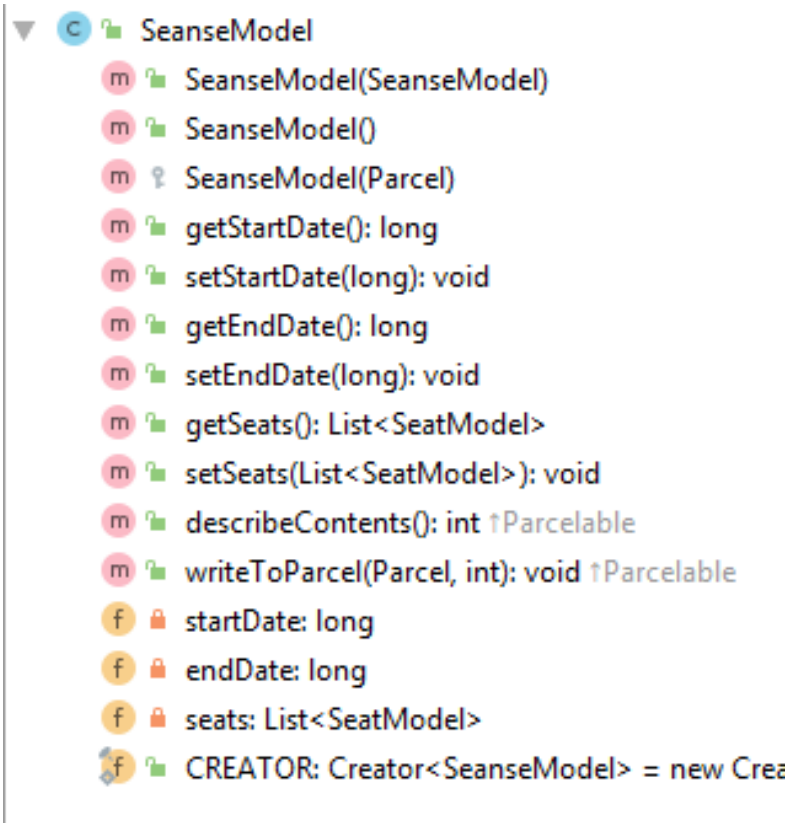

Рисунок 4.4 — Методи класу SeanseModel

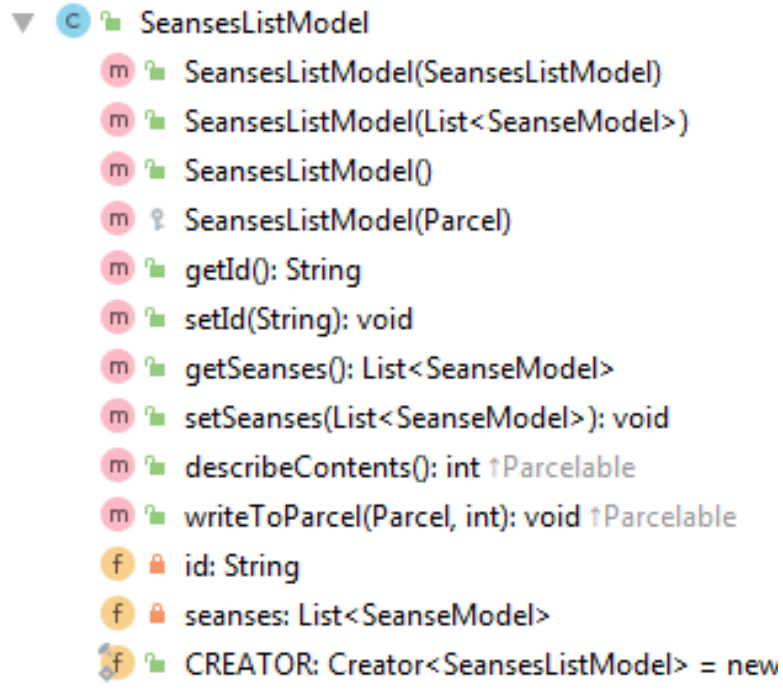

Рисунок 4.5 — Методи класу SeansesListModel

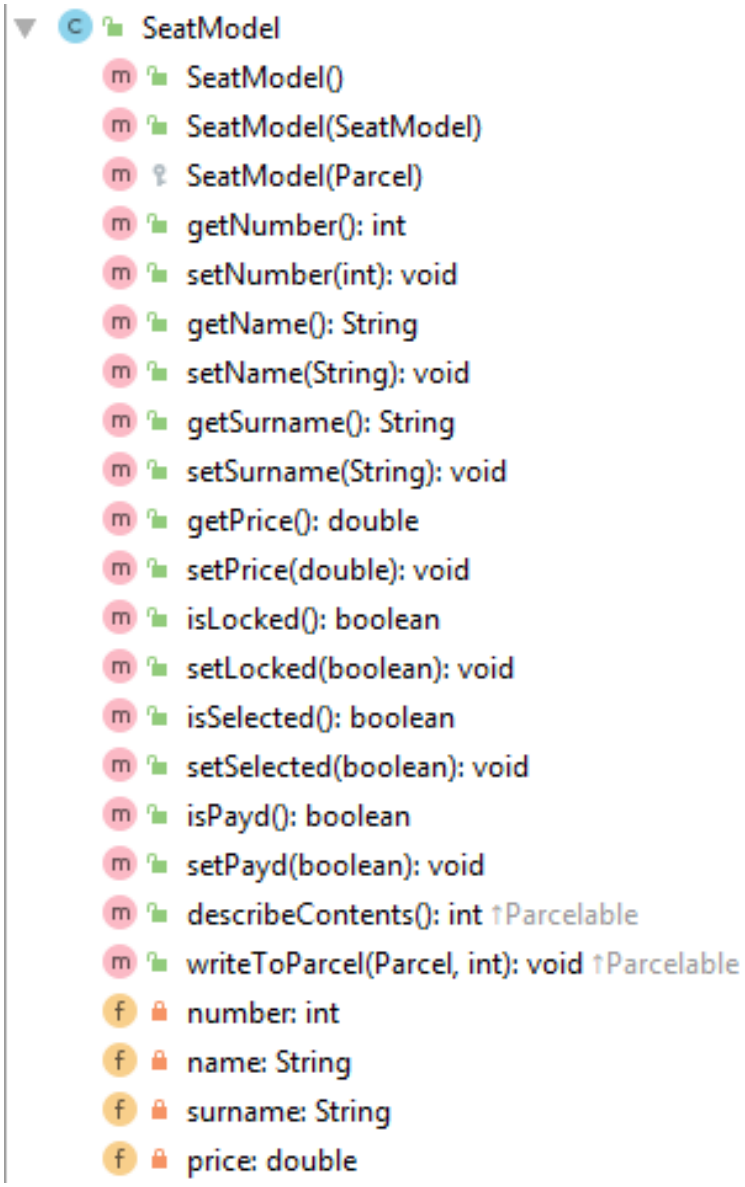

Рисунок 4.6 — Методи класу SeatModel

Таким чином, розроблені класи моделей дають можливість реалізувати роботу мобільного додатку Kino Booker щодо надання користувачеві різноманітної інформації.

4.3. Генератор фільмів, сеансів і місць

Генератор фільмів, сеансів і місць, як і вся програма, побудований за архітектурою MVP.

Моделлю в даному випадку виступає клас GeneratorActivity. Цей клас

створено виключно для взаємодії “адміністратор-система”. Методи, що використовуються в ньому, описують всі поля, потрібні для створення сторінок з фільмами, сеансами тощо.

Генератор запитує в адміністратора з додатку чи зчитує інформацію з бази даних таку, як:

— всю інформацію про фільм (назва, опис, картинка, трейлер тощо);

— дату і час сеансу;

— ціни.

Методи класу GeneratorActivity подано на рисунку 4.7.

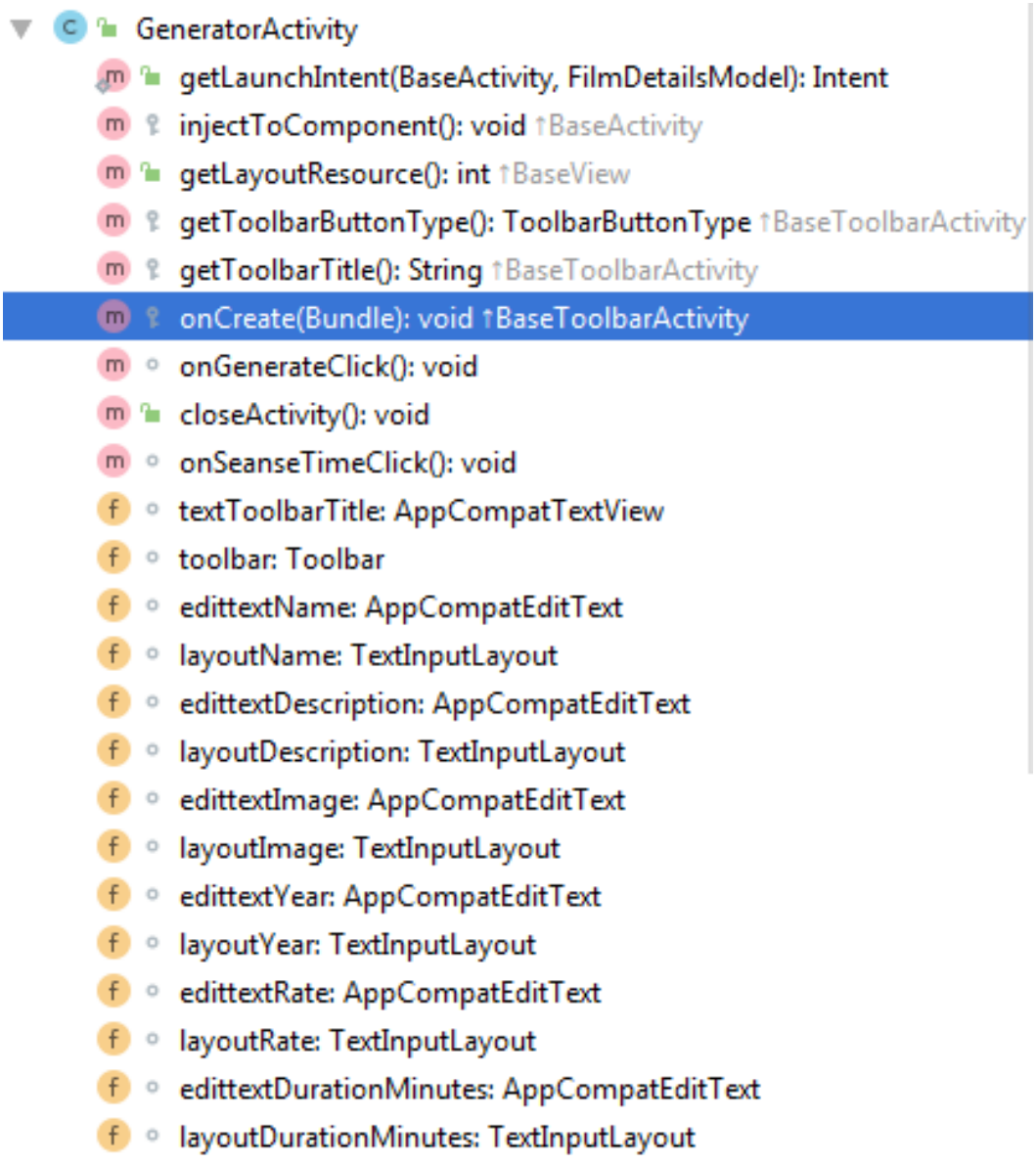

Рисунок 4.7 — Методи класу GeneratorActivity

Презентер генератора (клас GeneratorPresenter) звертається до класу GeneratorActivity та обробляє дані. Він задає модель фільму. Якщо її немає — то створюється новий. Якщо є — модель оновлюється.

Методи класу GeneratorPresenter показано на рисунку 4.8.

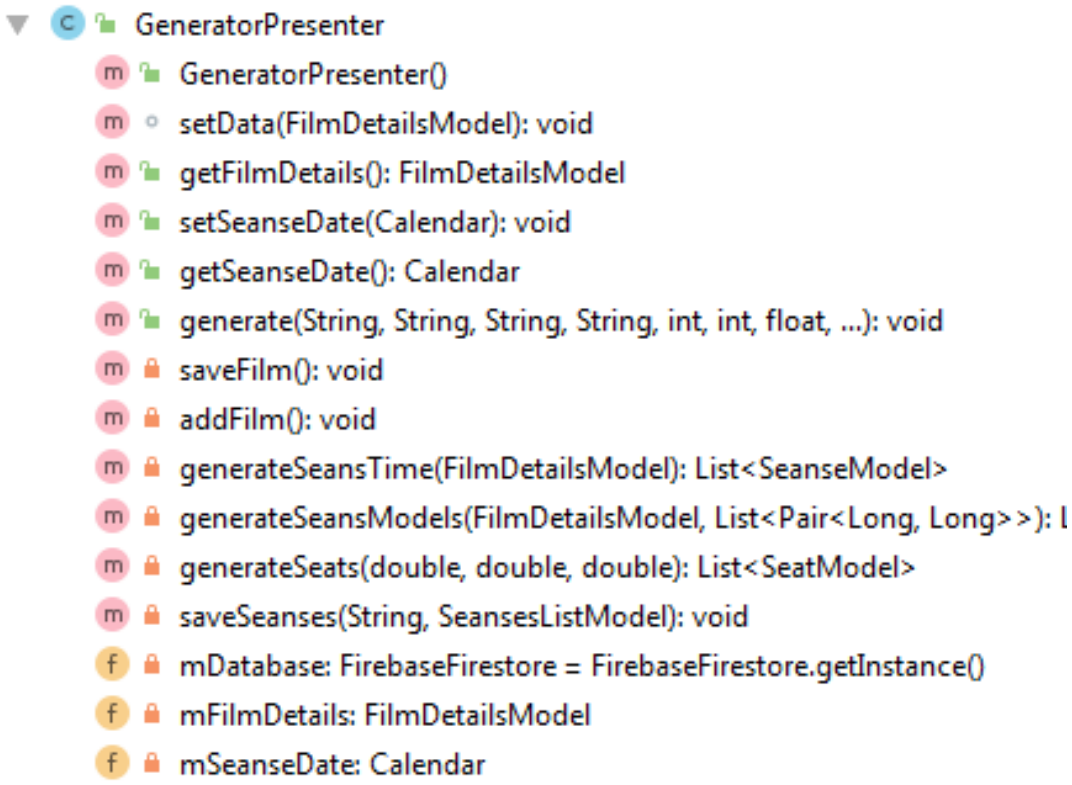

Рисунок 4.8 — Методи класу GeneratorPresenter

Після зчитування й перевірки всіх даних презентер генерує дані про фільм і зберігає їх на сервері. Якщо дані про фільм вже існують, то спрацьовує метод saveFilm, якщо ж ні — метод addFilm. Також після зчитування даних про тривалість фільму, генератор створює сеанси, розбиваючи їх згідно з тривалістю. Потім зчитує дані про ціни й генерує місця та записує їх на сервер.

4.4. Робота з базою даних

Вся інформація мобільного додатку Kino Booker зберігається на хмарному сервісі FireBase. Для адміністратора (менеджера) додаються права на створення

нових даних та редагування існуючих даних, які містяться в базі.

Бази даних Firebase, на відміну від mySQL, побудовані не таблицями, а деревом. Адміністратор може додати колекцію чи змінити її. Вигляд дерева колекції подано на рисунку 4.9.

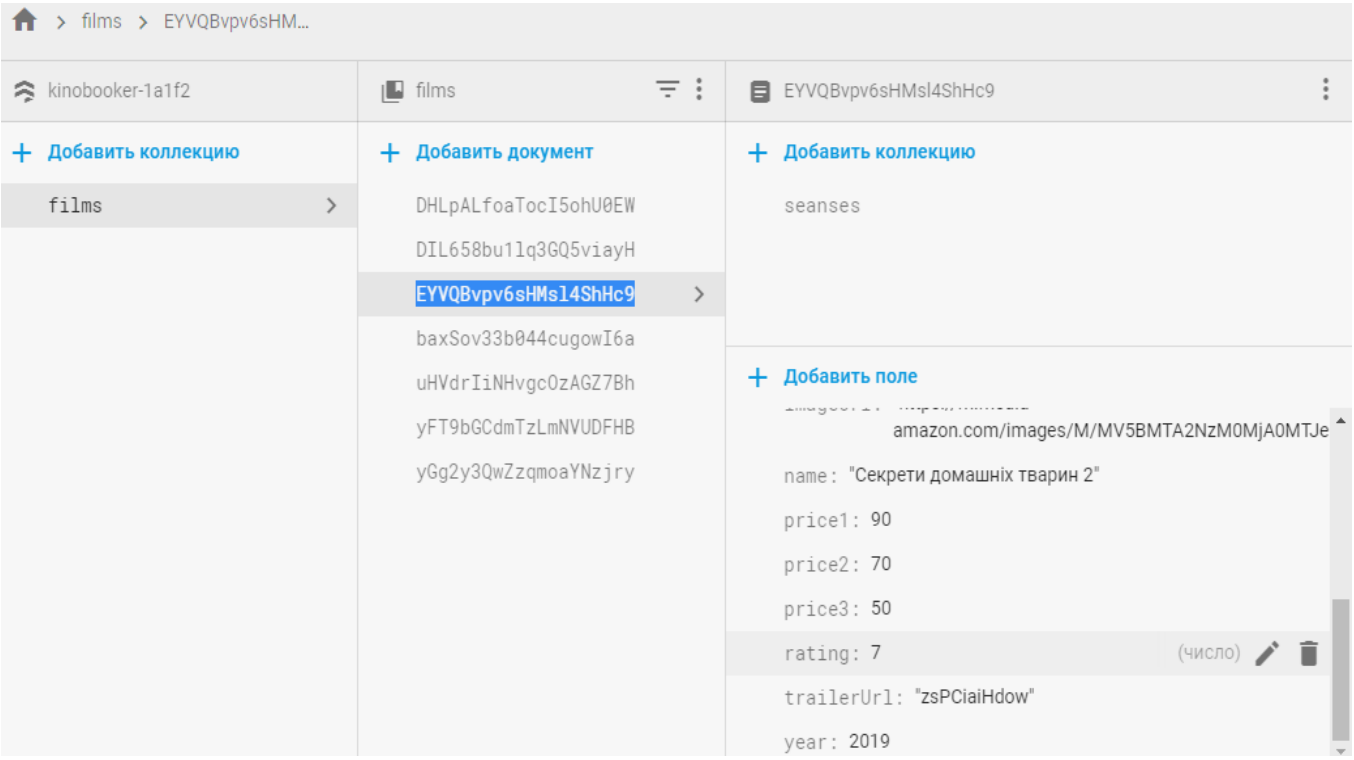

Рисунок 4.9 — Вигляд дерева колекцій на сервісі Firebase

Тут адміністратор може заповнити чи змінити основні дані про фільм та сеанси такі, як:

— поле description — опис фільму;

— поле duratioHours — тривалість фільму (повних годин);

— поле durationMinutes — залишкова тривалість;

— поле d — ідентифікатор фільму. Якщо залишити пустим (null), система створить його сама;

— поле imageUrl — посилання на постер до фільму;

— поле name — назва фільму;

— поля price1, price2, price3, … — цінова категорія місць;

— поле rating — оцінка imdb;

— поле trailerUrl — посилання на трейлер до фільму;

— поле year — рік виробництва.

Дані про бронювання чи купівлю квитка надходять у колекцію Seanses. Вигляд колекції місць показано на рисунку 4.10.

Рисунок 4.10 — Вигляд колекції місць

У колекції Seanses адміністратор може побачити, яке місце заброньоване, а яке вільне. Якщо воно заброньоване, то значення атрибута “locked” дорівнює true, якщо вільне — false. Також тут міститься інформація про номер місця і вартість квитка.

Кожне місце в кінозал на кожен сеанс і на кожен фільм має своє окреме місце у дереві бази даних. Воно має кілька атрибутів, наприклад: ідентифікатор (id), ціну (price), порядковий номер тощо. Як було сказано вище, щоб показати користувачеві, що місце зайняте, значення атрибута “locked” змінюється на true. А, щоб показати адміністраторові, що воно ще й оплачене, значення атрибута “payd” змінюється теж на true.

Загальну схему аналізу наявності вільного місця в залі кінотеатру і його відображення подано на рисунку 4.11.

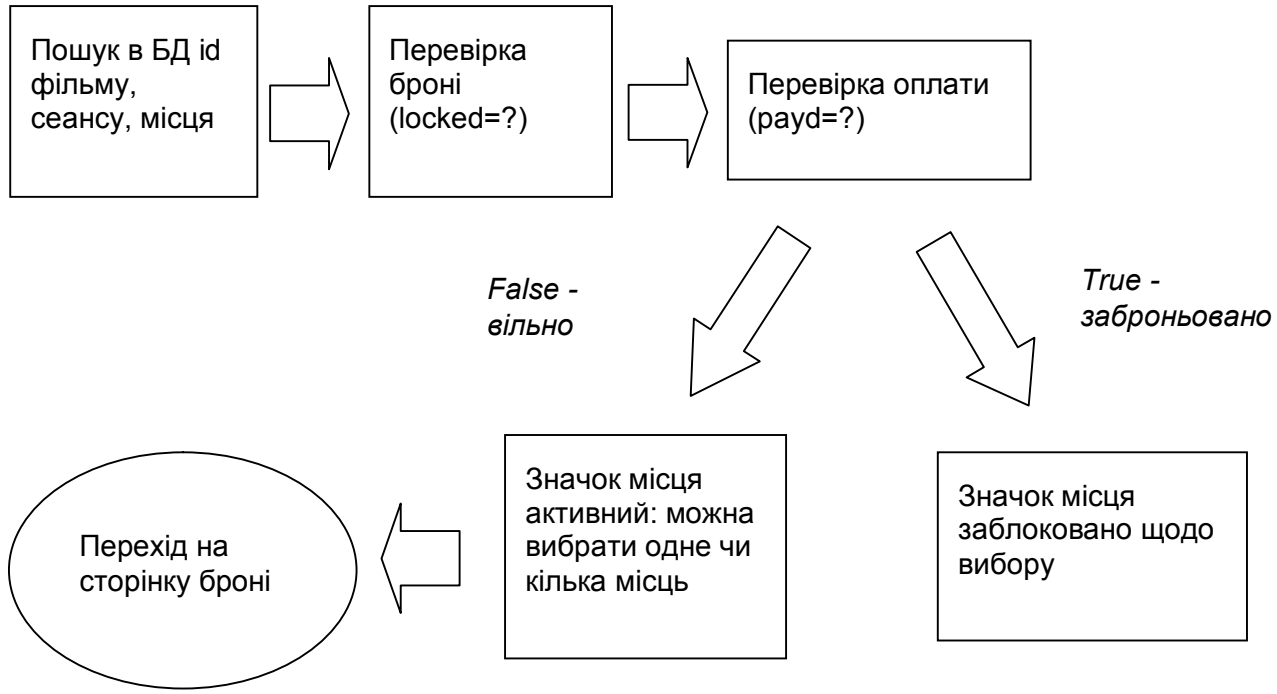

Рисунок 4.11 — Аналіз наявності вільного місця в кінотеатрі

Подання бази даних у формі дерева спрощує роботу адміністратора при додаванні чи зміні даних про фільм.

За допомогою адмін-панелі сервісу Firebase можна надати доступ до бази даних будь-якому користувачеві з правами власника, редактора чи глядача. Також можна додати права на окрему колекцію чи документ, що значно спрощує роботу адміністратора.

4.5. Тестування роботи системи

Тестування мобільного комплексу Kino Booker виконувалося на емуляторі Android-смартфонів AVD (Android Virtual Device), а також на смартфоні з версією операційної системи Android 9.0 (рисунки 4.12 — 4.14).

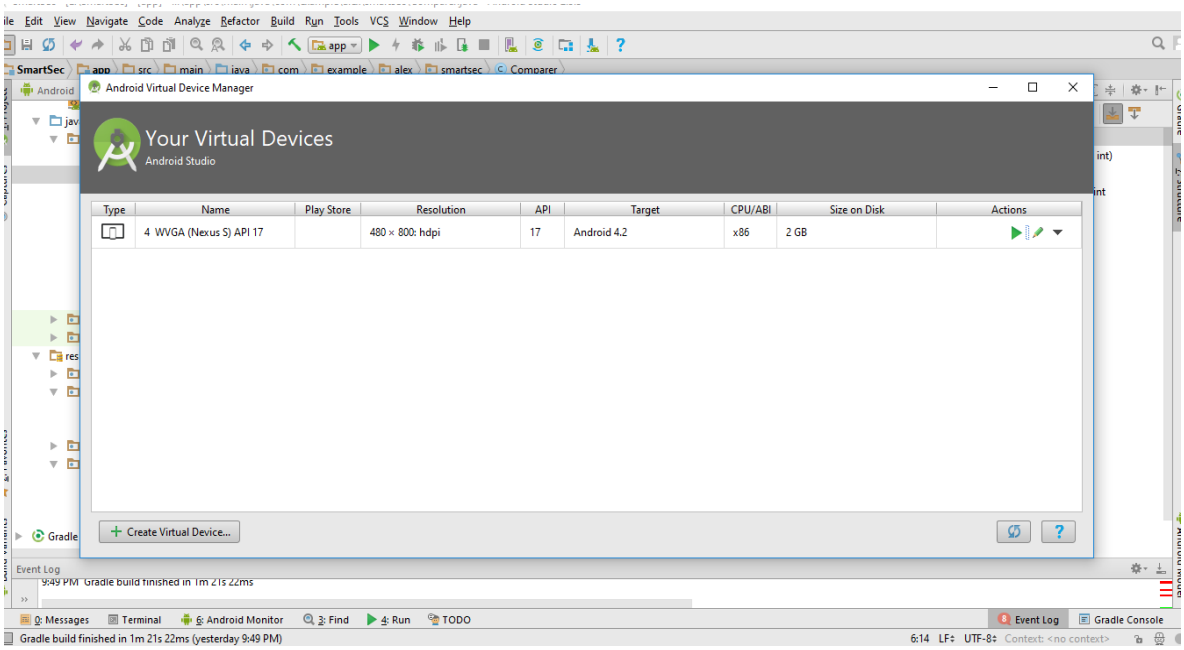

Рисунок 4.12 — Запуск емулятора Android-смартфонів

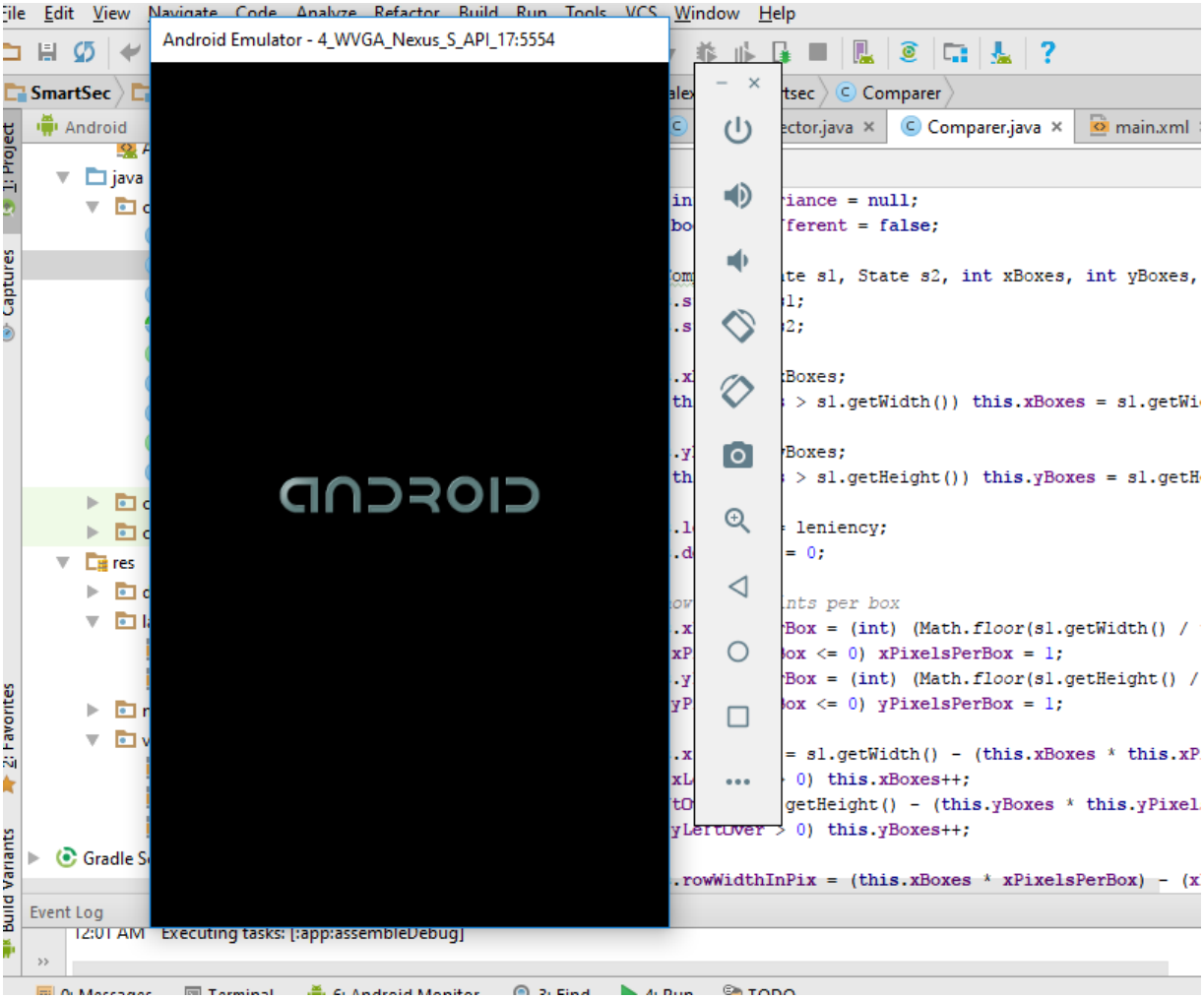

Рисунок 4.13 — Емуляція старту Android-смартфона

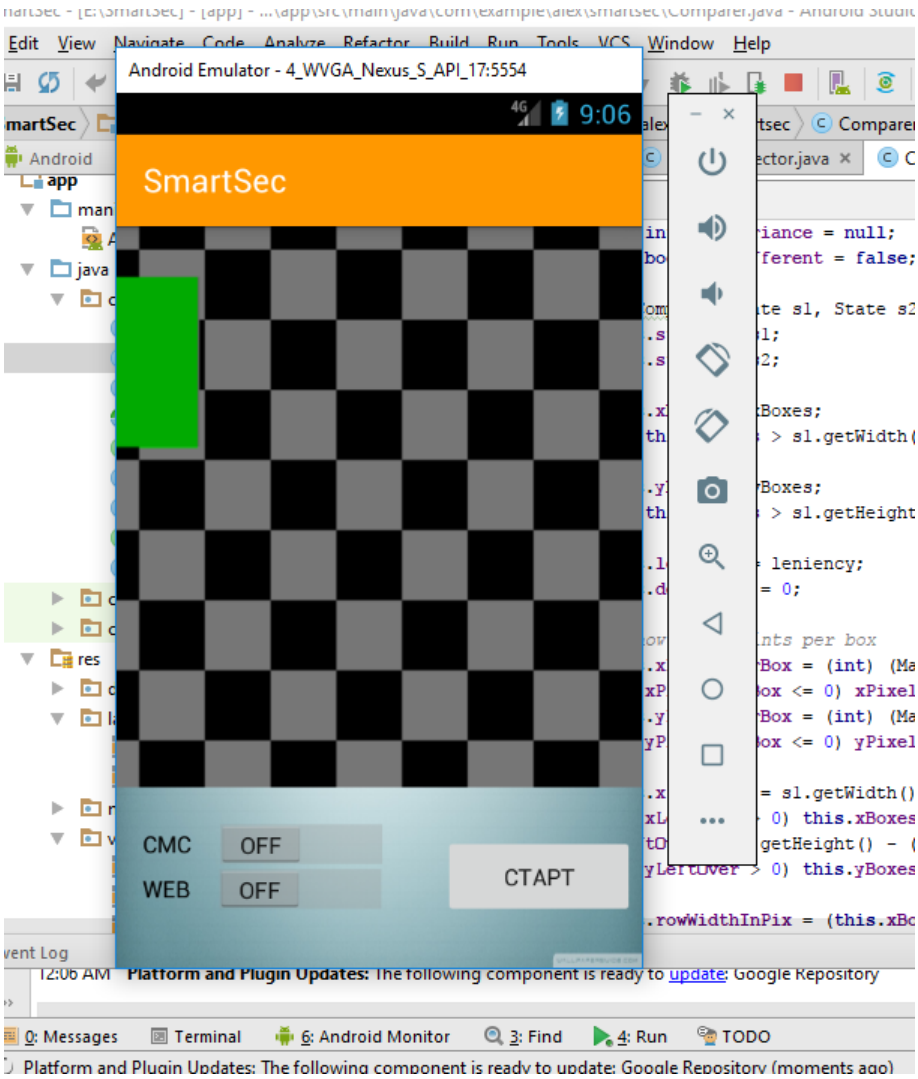

Рисунок 4.14 — Емуляція роботи додатку на Android-смартфоні

Використання емулятора Android-пристроїв дало можливість під час тестування мобільної системи бронювання квитків Kino Booker встановлювати різні технічні конфігурації смартфонів, версії операційної системи Android і роздільні здатності екранів.

Тестування на смартфоні з версією операційної системи Android 9.0 дало можливість випробувати систему Kino Booker в реальних умовах роботи.

5. МЕТОДИКА РОБОТИ КОРИСТУВАЧА З СИСТЕМОЮ БРОНЮВАННЯ КВИТКІВ У КІНОТЕАТРІ KINO BOOKER

Для забезпечення стабільної роботи створеної системи бронювання квитків Kino Booker потрібно дотримуватися основних вимог при її інсталяції щодо версії мобільної операційної системи і конфігурації мобільного пристрою, а також рекомендацій щодо використання.

5.1. Системні вимоги й інсталяція комплексу

Програму розроблено для мобільних пристроїв з операційною системою Android версій не нижче Android 5.0. Jelly Bean.

Мінімальні вимоги до конфігурації мобільного пристрою:

— оперативна пам’ять — 512 Мбайт і більше;

— вільне місце на карті пам’яті — не менше ніж 50 Мбайт;

— підключення до Інтернету на швидкості не менше ніж 16 Кбіт/с.

Для встановлення програми на смартфон треба скопіювати у внутрішню пам’ять смартфона файл kinobooker.apk, створений у середовищі розробки Android Studio.

Треба запусти цей файл, чим викличеться стандартний інсталятор операційної системи Android.

Під час інсталяції інсталятор запитує для програми підтвердження деяких прав на використання певних ресурсів смартфона.

Після підтвердження застосунок встановлюється в системі.

Крім цього, вже мають бути створені колекції фільмів і сеансів в базі даних Firebase. Це завдання виконує розробник системи.

5.2. Робота користувача з додатком Kino Booker

При запуску програми з’являється “екран привітання”. Це звичайна анімація, яка створюється класом Splash і використовується для того, щоб завантажити список фільмів, інформація про які зберігається в базі даних.

Екран привітання подано на рисунку 5.1.

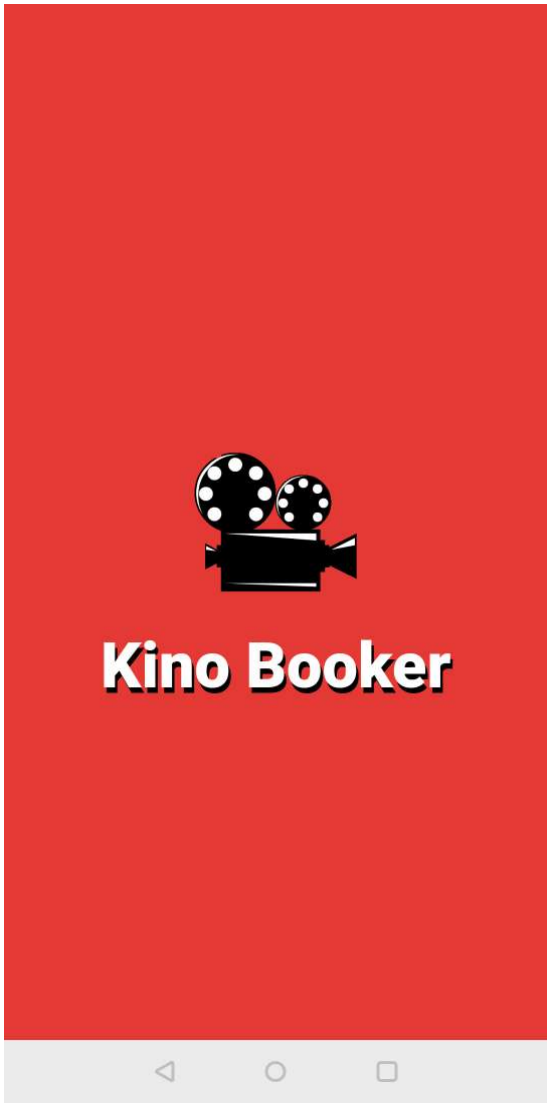

Рисунок 5.1 — Екран привітання

Завантажений список фільмів користувач побачить на екрані і зможе вибрати певний фільм.

Меню зі списком фільмів показано на рисунку 5.2.

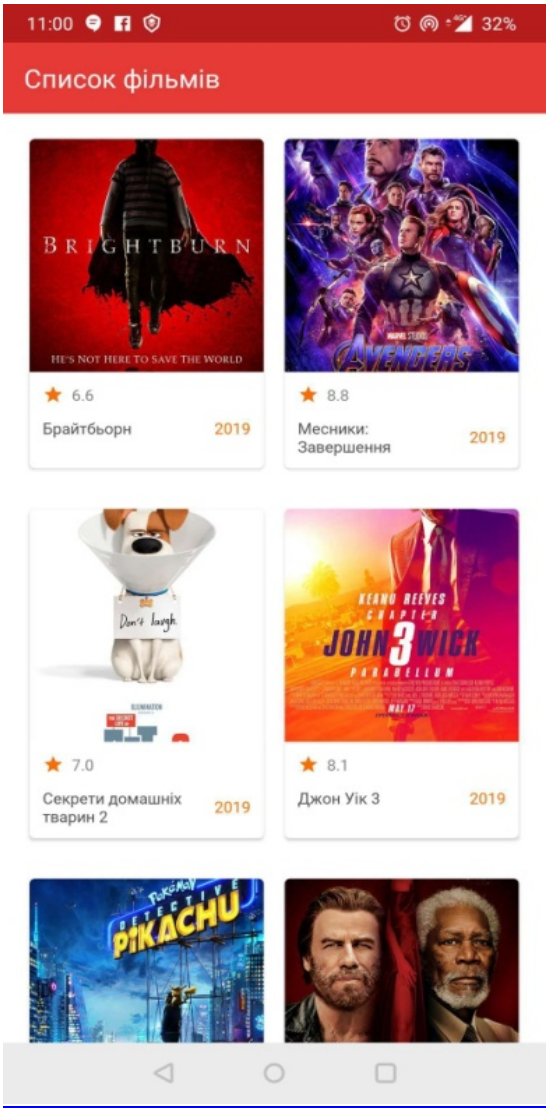

Рисунок 5.2 — Список фільмів

Після вибору фільму користувач потрапляє на його сторінку з постером, оцінкою, описом, тривалістю та кнопкою “замовити квитки”. Сторінка фільму зображена на рисунку 5.3.

Після натиснення на сторінці фільму кнопки “замовити квитки” користувач потрапляє в меню сеансів (рисунок 5.4), де вказано дату, час початку і час закінчення фільму, ціни квитків.

Після вибору вподобаного сеансу користувач потрапляє на сторінку кінозалу, де можна побачити всі місця за ціновою категорію, а також, які місця вільні, а які — заброньовані. Заброньовані місця є неактивними для натискання й позначені замком (рисунки 5.5 і 5.6).

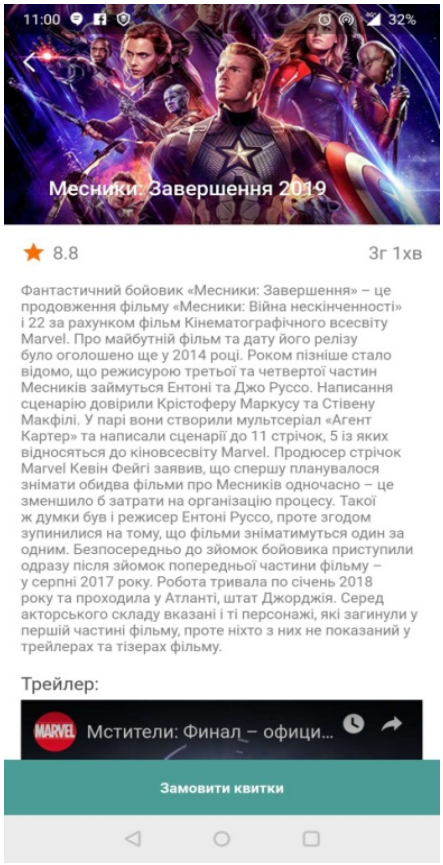

Рисунок 5.3 — Сторінка фільму

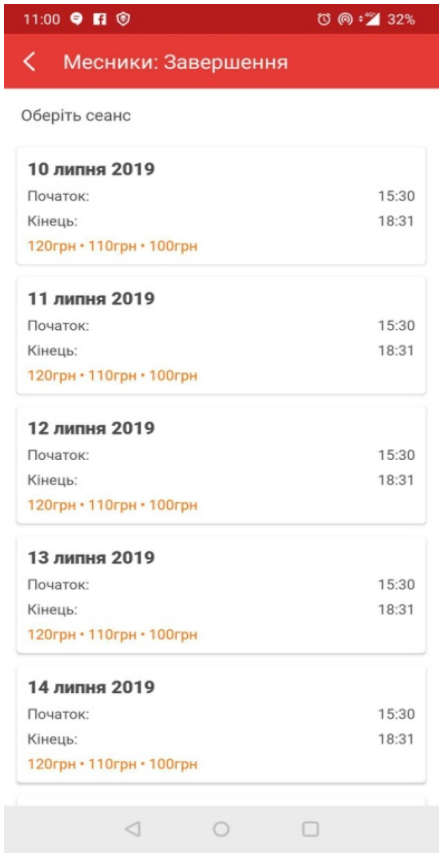

Рисунок 5.4 — Меню сеансів

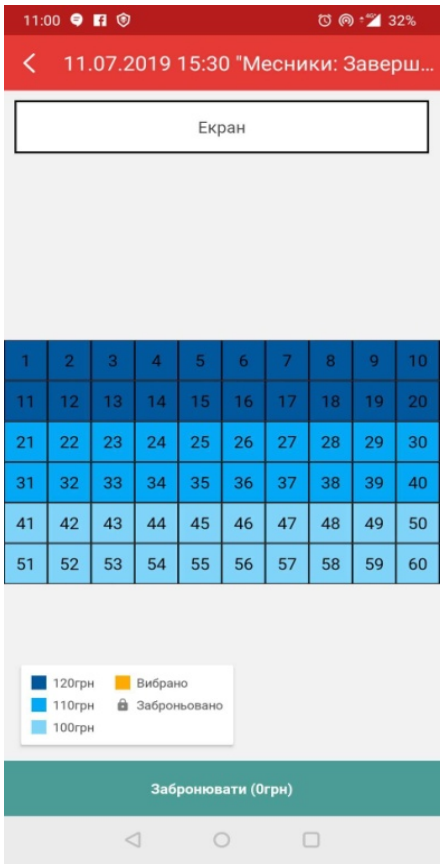

Рисунок 5.5 — Вільні місця (виділено кольором за ціновими категоріями)

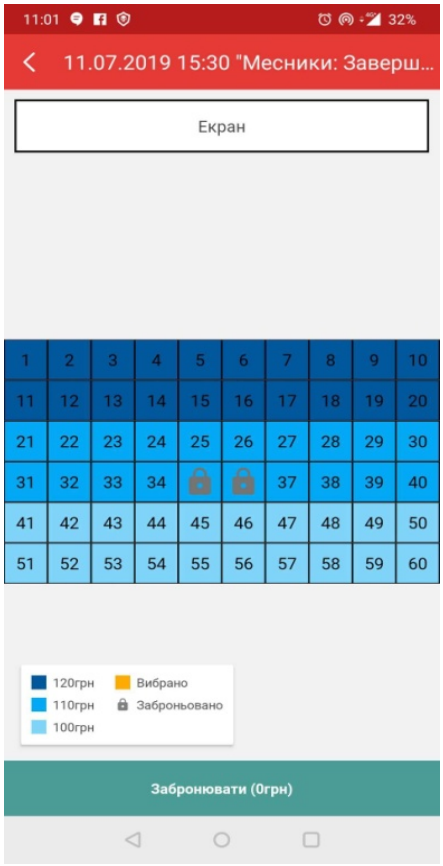

Рисунок 5.6 — Заброньовані місця (позначено замком)

Вибравши відповідне місце, користувач може забронювати чи одразу й оплатити його. При бронюванні чи оплаті потрібно ввести свої контактні дані, система перевірить, чи не залишилися поля порожніми. При оплаті система проводить валідацію банківської карти. Для тестування додано валідність для карти 4242 4242 4242 4242. Процес оплати зображено на рисунку 5.7.

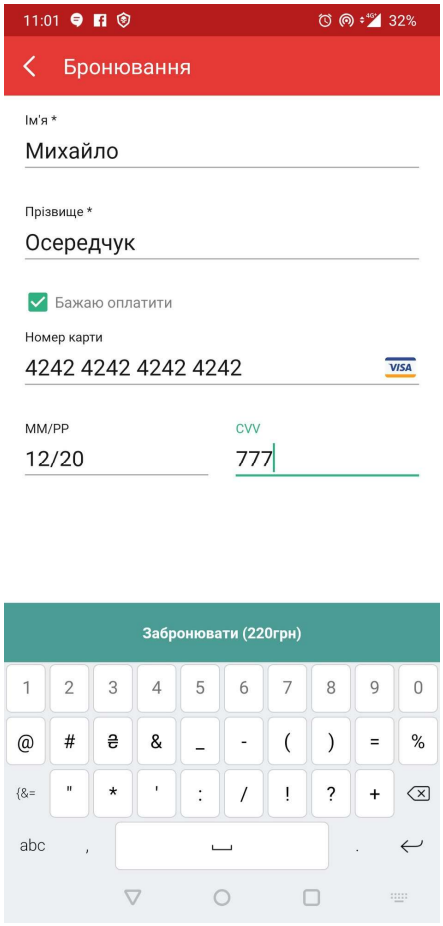

Рисунок 5.7 — Меню оплати квитка

Після резервування чи оплати користувач отримає сповіщення про успішну операцію, а дані будуть занесені в базу даних.

5.3. Робота адміністратора з додатком Kino Booker

Робота адміністратора із системою бронювання місць Kino Booker може виконуватися двома способами:

— з панелі керування базами даних FireBase;

— з мобільного додатку Kino Booker.

Щоб додати чи змінити дані про фільм, використовуючи панель керування базами даних FireBase, потрібно перейти в колекцію “films” і натиснути кнопку “Додати документ”, а потім ввести чи змінити всі потрібні дані.

Роботу адміністратора з базою даних у цьому випадку продемонстровано на рисунку 5.8.

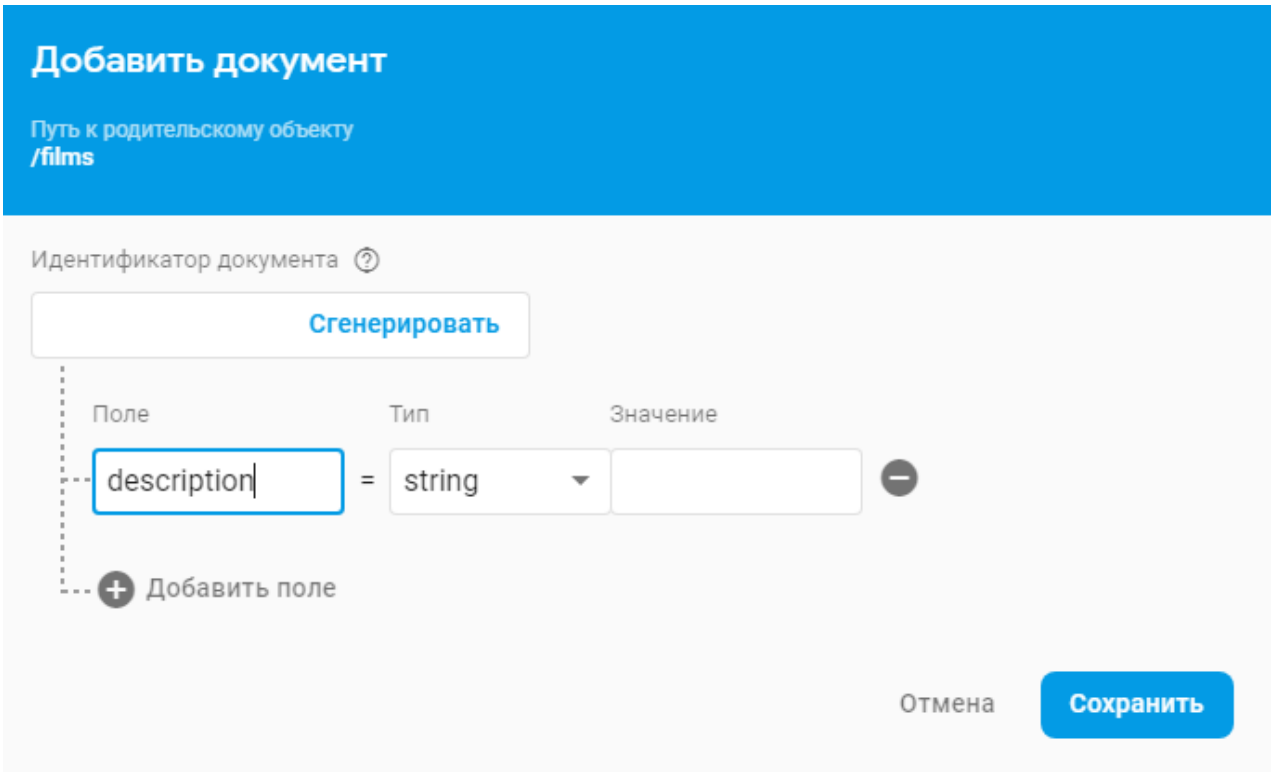

Рисунок 5.8 — Додавання адміністратором інформації про новий фільм

При роботі з використанням мобільного додатку Kino Booker, щоб викликати адмін-меню, адміністратор повинен натиснути й довго потримати натиснення на елементі потрібного фільму (рисунки 5.9 і 5.10).

У цьому адмін-меню адміністратор може додати дані про новий фільм, заповнивши всі обов’язкові поля, чи змінити введені раніше (тобто, наявні) дані про фільм, відредагувавши їх.

Генератор перевірить, чи інформація була внесена правильно. У разі, якщо все заповнено добре, відправить зміни на сервер.

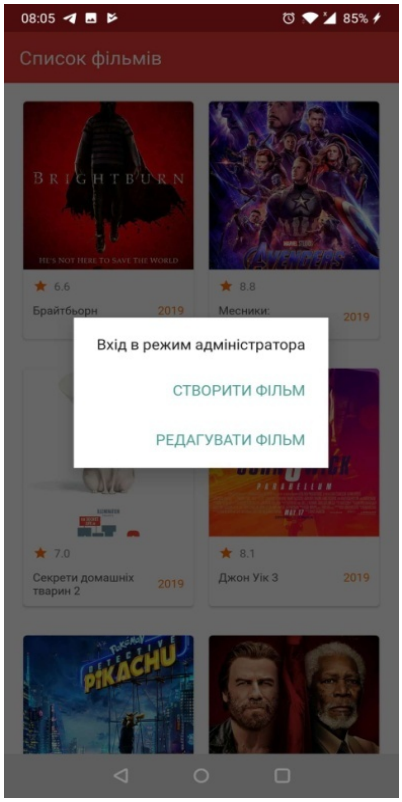

Рисунок 5.9 — Виклик режиму адміністратора

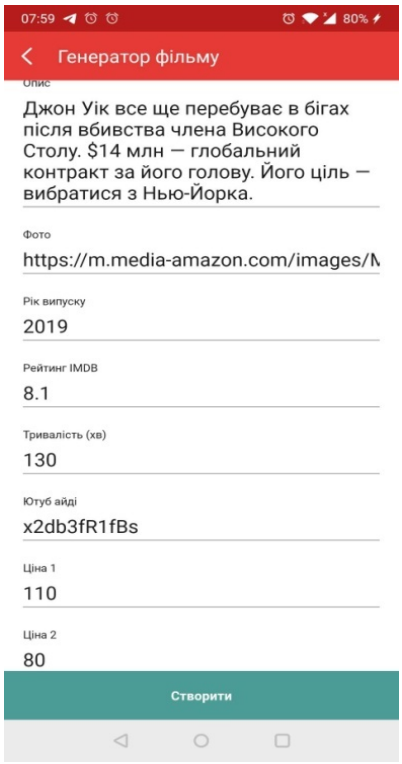

Рисунок 5.10 — Внесення змін адміністратором в базу даних

Таким чином, розроблена система надає широкі можливості роботи як звичайним користувачам, так і адміністратору.

ВИСНОВКИ

У ході виконання дипломної роботи було проведено аналіз існуючих програмних засобів для бронювання квитків у кінотеатрі з використання ОС Android, виявлено їхні недоліки, зроблено висновок про їхню недостатність і актуальність розробки нових застосунків для бронювання квитків.

Обґрунтовано вибір засобів реалізації: середовище розробки — Android Studio 3.4, мову програмування — Java, система керування базами даних Firebase.

Створено систему бронювання Kino Booker, яка працює на мобільних пристроях з операційною системою Android. Система забезпечує виконання ряду функцій, основними серед яких є: перегляд інформації про фільм, вибір вподобаного сеансу та місця, бронювання квитка чи покупка його.

Порівняно з аналогічними програмними рішеннями система бронювання Kino Booker має такі основні переваги: швидкість роботи, відсутність обов’язкової реєстрації, безкоштовність.

Потенційними користувачами програмного комплексу є звичайні користувачі, які мають мобільний пристрій з операційною системою Android версії не нижче 5.0.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Найбільший сервіс on-line продажу квитків у кіно [Електронний ресурс]. — Режим доступу: https://vkino.ua/ua/
2. Android-додаток “Вкіно — покупка квитків онлайн” [Електронний ресурс]. — Режим доступу: https://play.google.com/store/apps/details?id=ua.com.salesoft.vkino&hl=ua
3. Android-додаток “Планета Кіно” [Електронний ресурс]. — Режим доступу: https://play.google.com/store/apps/details?id=com.planet.imax
4. Android-додаток “SmartCinema” [Електронний ресурс]. — Режим доступу: https://play.google.com/store/apps/details?id=ua.kinoteatr.SmartCinema
5. Android Studio Release Notes [Електронний ресурс]. — Режим доступу: https://developer.android.com/studio/releases/index.html
6. Google добавила Kotlin в качестве официального языка программирования для Android [Електронний ресурс]. — Режим доступу: https://3dnews.ru/952400
7. Use Java 8 language features [Електронний ресурс]. — Режим доступу: https://developer.android.com/studio/write/java8-support
8. Wharton J. Android’s Java 9, 10, 11, and 12 Support [Електронний ресурс]. — Режим доступу: https://jakewharton.com/androids-java-9-10-11-and-12-support/
9. Android Studio system requirements [Електронний ресурс]. — Режим доступу: https://developer.android.com/studio
10. Learning Java [Електронний ресурс]. — NetBeans, 2015. — P. 3-16. — Режим доступу: https://netbeans.org/kb/articles/learn-java.html
11. Давыдов С.В. Intellij idea. Профессиональное программирование на Java / С.В. Давыдов, А.А. Ефимов. — СПб.: БХВ-Петербург, 2005. — 800 с.
12. Gosling J. The Java Language Environment: Contents. A White Paper [Електронний ресурс] / James Gosling, Henry McGilton // Sun Microsystems, May 1996. — Режим доступу: https://www.oracle.com/technetwork/java/langenv-140151.html

13. Android Captures Record 88% Share of Global Smartphone Shipments in Q3 2016 [Електронний ресурс]. — Режим доступу: https://www.linkedin.com/pulse/android-captures-record-88-share-global-smartphone-q3-stepanyan
14. Що таке OS Android [Електронний ресурс]. — Режим доступу: http://it-tehnolog.com/statti/scho-take-os-android
15. What is Android? About the platform [Електронний ресурс]. — Режим доступу: https://developer.android.com/guide/platform
16. Qualcomm and Google add 5G support to Android Q developer APIs [Електронний ресурс]. — Режим доступу: https://venturebeat.com/2019/05/07/qualcomm-and-google-add-5g-support-to-android-q-developer-apis/
17. Weber H. The top security threats to mobile users in 2012? Malware, sneaky ads and data thieves [Електронний ресурс] / Harrison Weber. — Режим доступу: https://thenextweb.com/apps/2011/12/14/lookout-reports-mobile-threats-for-2012/
18. Tamplin J. Firebase is Joining Google! [Електронний ресурс] / James Tamplin Firebase, Inc, 21.10.2014. — Режим доступу: https://firebase.googleblog.com/2014/10/firebase-is-joining-google.html
19. Google’s Cloud Firestore Lets You Focus On App Development [Електронний ресурс] /| Androidheadlines.com. AndroidHeadlines.com 2017-10-05. Режим доступу: https://www.androidheadlines.com/2017/10/googles-cloud-firestore-lets-focus-app-development.html
20. Firebase: Руководство [Електронний ресурс]. — Режим доступу: https://developer.android.com/distribute/best-practices/develop/build-with-firebase?hl=RU

ДОДАТОК А

Система автоматизованого бронювання квитків у кінотеатрі з використанням ОС Android

Специфікація

УКР.НТУУ“КПІ”. ТВз3104_19Б 81-1

Аркушів 2

2019

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ “КПІ”. ТВз3104_19Б 81-1	Записка.doc	Пояснювальна записка
Компоненти		
УКР.НТУУ “КПІ”. ТВз3104_19Б 12-1	FilmDetailsModel.java	Модель фільмів
УКР.НТУУ “КПІ”. ТВз3104_19Б 12-2	SeanseModel.java	Модель сеансів
УКР.НТУУ “КПІ”. ТВз3104_19Б 12-3	SeansesListModel.java	Модель списку сеансів
УКР.НТУУ “КПІ”. ТВз3104_19Б 12-4	SeatModel.java	Модель місць у кінозалі
УКР.НТУУ “КПІ”. ТВз3104_19Б 12-5	GeneratorActivity.java	Модель генератора
УКР.НТУУ “КПІ”. ТВз3104_19Б 12-6	GeneratorPresenter.java	Презентер генератора
УКР.НТУУ “КПІ”. ТВз3104_19Б 13-1	Опис програмного модуля	Модель та презентер генератора

ДОДАТОК Б

Система автоматизованого бронювання квитків у кінотеатрі з використанням ОС Android

Генератор фільмів, сеансів та місць у кінозалі

Текст програмного модуля

УКР.НТУУ “КПІ”. ТВз3104_19Б 12-1

Аркушів 18

2019

-2-

Модель

```
package com.kino.booker.ui.generator;

import android.app.DatePickerDialog;
import android.app.TimePickerDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.LinearLayout;
import android.widget.ScrollView;

import com.google.android.material.textfield.TextInputLayout;
import com.kino.booker.R;
import com.kino.booker.base.view.ToolbarButtonType;
import com.kino.booker.base.view.activity.BaseActivity;
import com.kino.booker.base.view.activity.BaseToolbarActivity;
import com.kino.booker.model.FilmDetailsModel;
import com.kino.booker.utils.DateUtils;
import com.kino.booker.utils.Extras;
import com.kino.booker.utils.card.CardSpan;

import java.util.Calendar;
import androidx.annotation.NonNull;
import androidx.appcompat.widget.AppCompatButton;
import androidx.appcompat.widget.AppCompatEditText;
import androidx.appcompat.widget.AppCompatTextView;
import androidx.appcompat.widget.Toolbar;
import butterknife.BindView;
import butterknife.OnClick;
```

```
public class GeneratorActivity extends BaseToolbarActivity<GeneratorPresenter>
implements GeneratorView {

    @BindView(R.id.text_toolbar_title) AppCompatTextView textToolbarTitle;
    @BindView(R.id.toolbar) Toolbar toolbar;
    @BindView(R.id.edittext_name) AppCompatEditText edittextName;
    @BindView(R.id.layout_name) TextInputLayout layoutName;
    @BindView(R.id.edittext_description) AppCompatEditText edittextDescription;
    @BindView(R.id.layout_description) TextInputLayout layoutDescription;
    @BindView(R.id.edittext_image) AppCompatEditText edittextImage;
    @BindView(R.id.layout_image) TextInputLayout layoutImage;
    @BindView(R.id.edittext_year) AppCompatEditText edittextYear;
    @BindView(R.id.layout_year) TextInputLayout layoutYear;
    @BindView(R.id.edittext_rate) AppCompatEditText edittextRate;
    @BindView(R.id.layout_rate) TextInputLayout layoutRate;
    @BindView(R.id.edittext_duration_minutes) AppCompatEditText
edittextDurationMinutes;
    @BindView(R.id.layout_duration_minutes) TextInputLayout
layoutDurationMinutes;
    @BindView(R.id.edittext_youtube_id) AppCompatEditText edittextYoutubeId;
    @BindView(R.id.layout_youtube_id) TextInputLayout layoutYoutubeId;
    @BindView(R.id.edittext_price_1) AppCompatEditText edittextPrice1;
    @BindView(R.id.layout_price_1) TextInputLayout layoutPrice1;
    @BindView(R.id.edittext_price_2) AppCompatEditText edittextPrice2;
    @BindView(R.id.layout_price_2) TextInputLayout layoutPrice2;
    @BindView(R.id.edittext_price_3) AppCompatEditText edittextPrice3;
    @BindView(R.id.layout_price_3) TextInputLayout layoutPrice3;
    @BindView(R.id.layout_root) LinearLayout layoutRoot;
    @BindView(R.id.scroll_view_root) ScrollView scrollViewRoot;
```

```
@BindView(R.id.button_generate) AppCompatButton buttonGenerate;
    @BindView(R.id.button_seanse_time) AppCompatButton buttonSeanseTime;

    public static Intent getLaunchIntent(final BaseActivity activity, FilmDetailsModel
filmDetails) {
        Intent intent = new Intent(activity, GeneratorActivity.class);
        intent.putExtra(Extras.FILM_DETAILS, filmDetails);
        return intent;
    }

    @Override
    protected void injectToComponent() {
        getActivityComponent().inject(this);
    }

    @Override
    public int getLayoutResource() {
        return R.layout.activity_generator;
    }

    @NonNull
    @Override
    protected ToolbarButtonType getToolbarButtonType() {
        return ToolbarButtonType.BACK;
    }

    @Override
    protected String getToolbarTitle() {
        return "Генератор фільму";
```

```
}

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (getIntent().getExtras() != null &&
getIntent().getExtras().getParcelable(Extras.FILM_DETAILS) != null) {
            buttonSeanseTime.setVisibility(View.GONE);
        }


getPresenter().setData(getIntent().getExtras().getParcelable(Extras.FILM_DETAILS)
);

        edittextName.setText(getPresenter().getFilmDetails().getName());
        edittextDescription.setText(getPresenter().getFilmDetails().getDescription());
        edittextImage.setText(getPresenter().getFilmDetails().getImageUrl());
        edittextYoutubeId.setText(getPresenter().getFilmDetails().getTrailerUrl());
        edittextYear.setText(getPresenter().getFilmDetails().getYear() + "");

edittextDurationMinutes.setText(getPresenter().getFilmDetails().getDurationHours()
* 60 + getPresenter().getFilmDetails().getDurationMinutes() + "");
        edittextRate.setText(getPresenter().getFilmDetails().getRating() + "");
        edittextPrice1.setText((int)
Math.round(getPresenter().getFilmDetails().getPrice1()) + "");
        edittextPrice2.setText((int)
Math.round(getPresenter().getFilmDetails().getPrice2()) + "");
        edittextPrice3.setText((int)
Math.round(getPresenter().getFilmDetails().getPrice3()) + "");
```

```
}

    @OnClick(R.id.button_generate)
    void onGenerateClick() {
        layoutName.setError(null);
        layoutDescription.setError(null);
        layoutImage.setError(null);
        layoutYoutubeId.setError(null);
        layoutYear.setError(null);
        layoutDurationMinutes.setError(null);
        layoutRate.setError(null);
        layoutPrice1.setError(null);
        layoutPrice2.setError(null);
        layoutPrice3.setError(null);

        if (edittextName.getText().toString().isEmpty()) {
            layoutName.setError("Має бути не пустим");
            return;
        }
        if (edittextDescription.getText().toString().isEmpty()) {
            layoutDescription.setError("Має бути не пустим");
            return;
        }
        if (edittextImage.getText().toString().isEmpty()) {
            layoutImage.setError("Має бути не пустим");
            return;
        }
        if (edittextYoutubeId.getText().toString().isEmpty()) {
            layoutYoutubeId.setError("Має бути не пустим");
```

```
return;
        }
        if (edittextYear.getText().toString().isEmpty()) {
            layoutYear.setError("Має бути не пустим");
            return;
        }
        if (edittextDurationMinutes.getText().toString().isEmpty()) {
            layoutDurationMinutes.setError("Має бути не пустим");
            return;
        }
        if (edittextRate.getText().toString().isEmpty()) {
            layoutRate.setError("Має бути не пустим");
            return;
        }
        if (edittextPrice1.getText().toString().isEmpty()) {
            layoutPrice1.setError("Має бути не пустим");
            return;
        }
        if (edittextPrice2.getText().toString().isEmpty()) {
            layoutPrice2.setError("Має бути не пустим");
            return;
        }
        if (edittextPrice3.getText().toString().isEmpty()) {
            layoutPrice3.setError("Має бути не пустим");
            return;
        }

        String name = edittextName.getText().toString().trim();
        String description = edittextDescription.getText().toString().trim();
```

```
String imageUrl = edittextImage.getText().toString().trim();
        String trailerUrl = edittextYoutubeId.getText().toString().trim();
        int year = Integer.parseInt(edittextYear.getText().toString().trim());
        int durationMinutes = Integer.parseInt(edittextDurationMinutes.getText().toString().trim());
        float rating = Float.parseFloat(edittextRate.getText().toString().trim());
        double price1 = Double.parseDouble(edittextPrice1.getText().toString().trim());
        double price2 = Double.parseDouble(edittextPrice2.getText().toString().trim());
        double price3 = Double.parseDouble(edittextPrice3.getText().toString().trim());

        if (getPresenter().getSeanseDate() == null) {
            showMessage(null, "Виберіть дату першого сеансу");
            return;
        }

        getPresenter().generate(name, description, imageUrl, trailerUrl, year, durationMinutes, rating, price1, price2, price3);
    }

    @Override
    public void closeActivity() {
        finish();
    }

    @OnClick(R.id.button_seanse_time)
    void onSeanseTimeClick() {
        final Calendar currentDate = Calendar.getInstance();
        Calendar date = Calendar.getInstance();
```

```
new DatePickerDialog(this, R.style.TimePickerDialogTheme, (view, year,
monthOfYear, dayOfMonth) -> {
        date.set(year, monthOfYear, dayOfMonth);
        new TimePickerDialog(GeneratorActivity.this,
R.style.TimePickerDialogTheme, (view1, hourOfDay, minute) -> {
            date.set(Calendar.HOUR_OF_DAY, hourOfDay);
            date.set(Calendar.MINUTE, minute);
            getPresenter().setSeanseDate(date);

buttonSeanseTime.setText(DateUtils.getDateStringByPattern(date.getTime(),
DateUtils.FULL_DATE_PATTERN_2));
        }, currentDate.get(Calendar.HOUR_OF_DAY),
currentDate.get(Calendar.MINUTE), true).show();
    }, currentDate.get(Calendar.YEAR), currentDate.get(Calendar.MONTH),
currentDate.get(Calendar.DATE)).show();
  }
}
```

Презентер

```
package com.kino.booker.ui.generator;

import android.util.Pair;

import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FirebaseFirestore;
import com.kino.booker.base.presenter.BasePresenter;
import com.kino.booker.di.scopes.ConfigPersistentScope;
```

```
import com.kino.booker.model.FilmDetailsModel;
import com.kino.booker.model.SeanseModel;
import com.kino.booker.model.SeansesListModel;
import com.kino.booker.model.SeatModel;
import com.kino.booker.utils.Constants;

import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;

import javax.inject.Inject;

@ConfigPersistentScope
public class GeneratorPresenter extends BasePresenter<GeneratorView> {
    private FirebaseFirestore mDatabase = FirebaseFirestore.getInstance();
    private FilmDetailsModel mFilmDetails;
    private Calendar mSeanseDate;

    @Inject
    public GeneratorPresenter() {

    }

    /**
     * Задає модель фільму. Якщо її немає - то створюється новий. Якщо є - то апдейтиться.
     *
     * @param filmDetails
     */
```

```
    void setData(FilmDetailsModel filmDetails) {
        if (filmDetails == null) {
            mFilmDetails = new FilmDetailsModel();
//            mFilmDetails.setName("Термінатор Генезис");
//            mFilmDetails.setDescription("Опис фільму термінатор");
//                                                mFilmDetails.setImageUrl("https://m.media-
amazon.com/images/M/MV5BMjM1NTc0NzE4OF5BMl5BanBnXkFtZTgwNDkyNj
Q1NTE@._V1_.jpg");
//            mFilmDetails.setTrailerUrl("jNU_jrPxs-0");
//            mFilmDetails.setYear(2015);
//            mFilmDetails.setDurationHours(2);
//            mFilmDetails.setDurationMinutes(6);
//            mFilmDetails.setRating(7);
//            mFilmDetails.setPrice1(110);
//            mFilmDetails.setPrice2(110);
//            mFilmDetails.setPrice3(110);
        } else {
            mFilmDetails = filmDetails;
            mSeanseDate = Calendar.getInstance();
        }
    }

    public FilmDetailsModel getFilmDetails() {
        return mFilmDetails;
    }

    public void setSeanseDate(Calendar seanseDate) {
        this.mSeanseDate = seanseDate;
    }
```

```
public Calendar getSeanseDate() {
    return mSeanseDate;
}

/**
 * Генерує фільм і зберігає його на сервері.
 */
/**
 * @param name
 * @param description
 * @param imageUrl
 * @param trailerUrl
 * @param year            - рік випуску
 * @param durationMinutes - тривалість (хв)
 * @param rating          - рейтинг IMDB
 * @param price1-         ціна 1
 * @param price2-         ціна 2
 * @param price3          - ціна 3
 */
public void generate(String name, String description, String imageUrl, String trailerUrl,
                     int year, int durationMinutes, float rating, double price1, double price2, double price3) {
    int hours = durationMinutes / 60;
    int minutes = durationMinutes % 60;

    mFilmDetails.setName(name);
    mFilmDetails.setDescription(description);
```

```
        mFilmDetails.setImageUrl(imageUrl);
        mFilmDetails.setTrailerUrl(trailerUrl);
        mFilmDetails.setYear(year);
        mFilmDetails.setDurationHours(hours);
        mFilmDetails.setDurationMinutes(minutes);
        mFilmDetails.setRating(rating);
        mFilmDetails.setPrice1(price1);
        mFilmDetails.setPrice2(price2);
        mFilmDetails.setPrice3(price3);

        if (mFilmDetails.getId() == null) {
            addFilm();
        } else {
            saveFilm();
        }
    }

    /**
     * Зберігає редагований фільм
     */
    private void saveFilm() {
        getView().showProgress(true);
        DocumentReference ref =
mDatabase.document(Constants.COLLECTION_FILMS + "/" +
mFilmDetails.getId());
        ref.set(mFilmDetails).addOnSuccessListener(documentReference -> {
            getView().showProgress(false);
            getView().closeActivity();
        }).addOnFailureListener(e -> {
```

```
            getView().showProgress(false);
            getView().showError("Сталася помилка", "" + e.getMessage());
        });
    }

    /**
     * Додає новий фільм
     */
    private void addFilm() {
        getView().showProgress(true);
        SeansesListModel seansesList = new
SeansesListModel(generateSeansTime(mFilmDetails));
        mDatabase.collection(Constants.COLLECTION_FILMS)
                .add(mFilmDetails)
                .addOnSuccessListener(documentReference -> {
                    mFilmDetails.setId(documentReference.getId());
                    saveSeanses(mFilmDetails.getId(), seansesList);
                })
                .addOnFailureListener(e -> {
                    getView().showProgress(false);
                    getView().showError("Сталася помилка", "" + e.getMessage());
                });
    }

    /**
     * Генерує час сеансівю
     *
     * @param film
     * @return
```

```
     */
    private List<SeanseModel> generateSeansTime(FilmDetailsModel film) {
        Calendar start = (Calendar) mSeanseDate.clone();

        List<Pair<Long, Long>> times = new ArrayList<>();
        for (int i = 0; i < 10; i++) {
            long first = start.getTimeInMillis();
            start.add(Calendar.HOUR_OF_DAY, film.getDurationHours());
            start.add(Calendar.MINUTE, film.getDurationMinutes());
            long second = start.getTimeInMillis();
            times.add(new Pair<>(first, second));
            start.add(Calendar.DAY_OF_MONTH, 1);
            start.set(Calendar.HOUR_OF_DAY,
mSeanseDate.get(Calendar.HOUR_OF_DAY));
            start.set(Calendar.MINUTE, mSeanseDate.get(Calendar.MINUTE));
        }
        return generateSeansModels(film, times);
    }

    /**
     * Генерує сеанси
     *
     * @param film
     * @param times
     * @return
     */
    private List<SeanseModel> generateSeansModels(FilmDetailsModel film,
List<Pair<Long, Long>> times) {
        List<SeanseModel> seansesList = new ArrayList<>();
```

```
        for (Pair<Long, Long> time : times) {
            SeanseModel seansModel = new SeanseModel();
            seansModel.setStartDate(time.first);
            seansModel.setEndDate(time.second);
            seansModel.setSeats(generateSeats(film.getPrice1(), film.getPrice2(),
film.getPrice3()));
            seansesList.add(seansModel);
        }
        return seansesList;
    }

    /**
     * Генерує місця
     *
     * @param price1
     * @param price2
     * @param price3
     * @return
     */
    private List<SeatModel> generateSeats(double price1, double price2, double
price3) {
        List<SeatModel> seats = new ArrayList<>();
        for (int i = 1; i < 21; i++) {
            SeatModel seatModel = new SeatModel();
            seatModel.setNumber(i);
            seatModel.setPrice(price1);
            seatModel.setLocked(false);
            seats.add(seatModel);
        }
```

```
        for (int i = 21; i < 41; i++) {
            SeatModel seatModel = new SeatModel();
            seatModel.setNumber(i);
            seatModel.setPrice(price2);
            seatModel.setLocked(false);
            seats.add(seatModel);
        }
        for (int i = 41; i < 61; i++) {
            SeatModel seatModel = new SeatModel();
            seatModel.setNumber(i);
            seatModel.setPrice(price3);
            seatModel.setLocked(false);
            seats.add(seatModel);
        }
        return seats;
    }

    /**
     * Зберігає сеанси
     *
     * @param filmId
     * @param seansesList
     */
    private void saveSeanses(String filmId, SeansesListModel seansesList) {
        mDatabase.collection(Constants.COLLECTION_FILMS + "/" + filmId + "/" +
Constants.COLLECTION_SEANSES)
                .add(seansesList)
                .addOnSuccessListener(documentReference -> {
                    getView().showProgress(false);
```

```
                getView().closeActivity();
            })
            .addOnFailureListener(e -> {
                getView().showProgress(false);
                getView().showError("Сталася помилка", "" + e.getMessage());
            });
    }
}
```

ДОДАТОК В

Система автоматизованого бронювання квитків у кінотеатрі з використанням ОС Android

Генератор фільмів, сеансів та місць у кінозалі

Опис програмного модуля

УКР.НТУУ “КПІ”. ТВз3104_19Б 13-1

Аркушів 7

2019

АНОТАЦІЯ

Розроблений генератор фільмів, сеансів та місць у кінозалі призначений для зчитування, обробки даних з бази даних чи програми від адміністратора, обробка інформації та виведення інформації, сприятливої для перегляду користувачем, у мобільний додаток

Як вхідні дані генератор отримує дані з бази даних.

Як результат генератор виводить на екран додатка сторінки з фільмами, сеансами та місцями у кінозалі.

Мова програмної реалізації — Java.

Середовище розробки — Android Studio 3.4.

ЗМІСТ

1. ЗАГАЛЬНІ ВІДОМОСТІ

Генератор складається з моделі «GeneratorActivity» та презентера «GeneratorPresenter». Мова програмної реалізації — Java. Середовище розробки — Android Studio 3.4.

Генератор є складовою частиною програмного продукту, що об’єднує всі існуючі конфігурації й застосовує їх.

Генератор працює з інформацією, яку отримує з бази даних та вирішує, яку з неї і в якому вигляді подавати у застосунок смартфона.

2. ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ ГЕНЕРАТОРА

Функціональним призначенням генератора є зв’язок між моделями фільмів, сеансів та ін. та обробкою інформації з бази даних і подання їх у вигляді додатку.

Генератор за допомогою звернення до інших методів знаходить розв’язання поставленої задачі. Він викликає виконання таких методів:

— setData — задає модель фільму;

— generate — генерує фільм та зберігає його на сервер;

— saveFilm — зберігає відредагований фільм;

— addFilm — додає новий фільм;

— generateSeansTime — генерує час сеансів;

— generateSeansModels — генерує модель сеансів;

— generateSeats — генерує місця;

— saveSeanses — зберігає сеанси.

3. ВЗАЄМОДІЯ ГЕНЕРАТОРА З ІНШИМИ КОМПОНЕНТАМИ СИСТЕМИ

Генератор взаємодіє з моделями та виглядами фільмів, сеансів, місць. Ці моделі містяться у папках model та ui (user interfaces)

4. ВХІДНІ ТА ВИХІДНІ ДАНІ

Як вхідні дані генератор отримує дані з бази даних та мобільного додатку.

Як результат генератор повертає вигляд сторінок фільмів, сеансів та місць у кінозалі.